QUANTUM ALGORITHMS

A Survey of Applications and End-to-end Complexities

ALEXANDER M. DALZELL, SAM MCARDLE, ET AL.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Quantum Algorithms

The 1994 discovery of Shor's quantum algorithm for integer factorization—an important practical problem in the area of cryptography—demonstrated quantum computing's potential for real-world impact. Since then, researchers have worked intensively to expand the list of practical problems that quantum algorithms can solve effectively.

This book surveys the fruits of this effort, covering proposed quantum algorithms for concrete problems in many application areas, including quantum chemistry, optimization, finance, and machine learning. For each quantum algorithm considered, the book clearly states the problem being solved and the full computational complexity of the procedure, making sure to account for the contribution from all the underlying primitive ingredients. Separately, the book provides a detailed, independent summary of the most common algorithmic primitives. It has a modular, encyclopedic format to facilitate navigation of the material and to provide a quick reference for designers of quantum algorithms and quantum computing researchers.

ALEXANDER M. DALZELL is a Research Scientist at the Amazon Web Services Center for Quantum Computing.

SAM MCARDLE is a Research Scientist at the Amazon Web Services Center for Quantum Computing.

MARIO BERTA is a Professor of Physics at the Institute for Quantum Information at RWTH Aachen University and a Visiting Reader in the Department of Computing at Imperial College London.

PRZEMYSŁAW BIENIAS is a Research Scientist at the Amazon Web Services Center for Quantum Computing.

CHI-FANG CHEN is a Postdoctoral Scholar in the Department of Electrical Engineering and Computer Sciences at the University of California, Berkeley.

ANDRÁS GILYÉN is a Research Fellow in the Department of Probability and Statistics at the HUN-REN Alfréd Rényi Institute of Mathematics.

CONNOR T. HANN is a Senior Research Scientist at the Amazon Web Services Center for Quantum Computing.

MICHAEL J. KASTORYANO is an Associate Professor of Quantum Computing at the University of Copenhagen and an Amazon Visiting Academic at the Amazon Web Services Center for Quantum Computing.

EMIL T. KHABIBOULLINE is a National Resource Council Postdoctoral Associate at the Joint Center for Quantum Information and Computer Science (QuICS) of the University of Maryland, College Park, and the National Institute of Standards and Technology.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ALEKSANDER KUBICA is an Assistant Professor in the Department of Applied Physics at Yale University.

GRANT SALTON is a Senior Research Scientist at the Amazon Quantum Solutions Lab and the Amazon Web Services Center for Quantum Computing.

SAMSON WANG is a Postdoctoral Scholar at the Institute for Quantum Information and Matter at Caltech.

FERNANDO G. S. L. BRANDÃO is Director of Quantum Applications at the Amazon Web Services Center for Quantum Computing and Bren Professor of Theoretical Physics at Caltech.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Quantum Algorithms

A Survey of Applications and End-to-end Complexities

ALEXANDER M. DALZELL
AWS Center for Quantum Computing

SAM McARDLE
AWS Center for Quantum Computing

MARIO BERTA
RWTH Aachen University and Imperial College London

PRZEMYSŁAW BIENIAS
AWS Center for Quantum Computing

CHI-FANG CHEN
University of California, Berkeley

ANDRÁS GILYÉN
HUN-REN Alfréd Rényi Institute of Mathematics

CONNOR T. HANN
AWS Center for Quantum Computing

MICHAEL J. KASTORYANO
University of Copenhagen and AWS Center for Quantum Computing

EMIL T. KHABIBOULLINE
National Institute of Standards and Technology and University of Maryland, College Park

ALEKSANDER KUBICA
Yale University

GRANT SALTON
Amazon Quantum Solutions Lab and AWS Center for Quantum Computing

SAMSON WANG
Caltech

FERNANDO G. S. L. BRANDÃO
AWS Center for Quantum Computing and Caltech

CAMBRIDGE UNIVERSITY PRESS

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Shaftesbury Road, Cambridge CB2 8EA, United Kingdom

One Liberty Plaza, 20th Floor, New York, NY 10006, USA

477 Williamstown Road, Port Melbourne, VIC 3207, Australia

314–321, 3rd Floor, Plot 3, Splendor Forum, Jasola District Centre,
New Delhi – 110025, India

103 Penang Road, #05–06/07, Visioncrest Commercial, Singapore 238467

Cambridge University Press is part of Cambridge University Press & Assessment,
a department of the University of Cambridge.

We share the University's mission to contribute to society through the pursuit of
education, learning and research at the highest international levels of excellence.

www.cambridge.org
Information on this title: www.cambridge.org/9781009639644

DOI: 10.1017/9781009639651

© Amazon Technologies, Inc., California Institute of Technology, and Alfréd Rényi
Institute of Mathematics 2025

This publication is in copyright. Subject to statutory exception and to the provisions
of relevant collective licensing agreements, no reproduction of any part may take
place without the written permission of Cambridge University Press & Assessment.

An online version of this work is published at doi.org/10.1017/9781009639651 under
a Creative Commons Open Access license CC-BY-NC-ND 4.0 which permits re-use,
distribution and reproduction in any medium for non-commercial purposes providing
appropriate credit to the original work is given. You may not distribute derivative
works without permission. To view a copy of this license, visit
https://creativecommons.org/licenses/by-nc-nd/4.0

When citing this work, please include a reference to the DOI 10.1017/9781009639651

First published 2025

Cover image: ArtHead/Shutterstock.com

A catalogue record for this publication is available from the British Library

*A Cataloging-in-Publication data record for this book is available from the Library
of Congress*

ISBN 978-1-009-63964-4 Hardback
ISBN 978-1-009-63966-8 Paperback

Cambridge University Press & Assessment has no responsibility for the persistence
or accuracy of URLs for external or third-party internet websites referred to in this
publication and does not guarantee that any content on such websites is, or will
remain, accurate or appropriate.

For EU product safety concerns, contact us at Calle de José Abascal, 56, 1°, 28003
Madrid, Spain, or email eugpsr@cambridge.org.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Contents

Preface *page* ix

PART I AREAS OF APPLICATION 1

1 Condensed matter physics 4
1.1 Fermi–Hubbard model 5
1.2 Spin models 15
1.3 SYK model 24

2 Quantum chemistry 29
2.1 Simulating electrons in molecules and materials 29
2.2 Simulating vibrations in molecules and materials 45

3 Nuclear and particle physics 51
3.1 Quantum field theories 51
3.2 Nuclear physics 57

4 Combinatorial optimization 62
4.1 Search algorithms à la Grover 63
4.2 Beyond quadratic speedups in exact combinatorial optimization 68

5 Continuous optimization 78
5.1 Zero-sum games: Computing Nash equilibria 78
5.2 Conic programming: Solving LPs, SOCPs, and SDPs 82
5.3 General convex optimization 90
5.4 Nonconvex optimization: Escaping saddle points and finding local minima 94

6 Cryptanalysis 98
6.1 Breaking cryptosystems 99

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

6.2 Weakening cryptosystems 107
7 Solving differential equations 111
8 Finance 130
8.1 Portfolio optimization 132
8.2 Monte Carlo methods: Option pricing 140
9 Machine learning with classical data 148
9.1 Quantum machine learning via quantum linear algebra 149
9.2 Quantum machine learning via energy-based models 160
9.3 Tensor PCA 169
9.4 Topological data analysis 173
9.5 Quantum neural networks and quantum kernel methods 179

PART II QUANTUM ALGORITHMIC PRIMITIVES 185
10 Quantum linear algebra 188
10.1 Block-encodings 189
10.2 Manipulating block-encodings 195
10.3 Quantum signal processing 198
10.4 Qubitization 201
10.5 Quantum singular value transformation 205
11 Hamiltonian simulation 210
11.1 Product formulas 211
11.2 qDRIFT 216
11.3 Taylor and Dyson series (linear combination of unitaries) 218
11.4 Quantum signal processing / quantum singular value transformation 222
12 Quantum Fourier transform 225
13 Quantum phase estimation 228
14 Amplitude amplification and estimation 235
14.1 Amplitude amplification 235
14.2 Amplitude estimation 238
15 Gibbs sampling 243
16 Quantum adiabatic algorithm 250
17 Loading classical data 255
17.1 Quantum random access memory 255
17.2 Preparing quantum states from classical data 259
17.3 Block-encoding dense matrices of classical data 266

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

18 Quantum linear system solvers 270
19 Quantum gradient estimation 276
20 Variational quantum algorithms 281
21 Quantum tomography 286
22 Quantum interior point methods 291
23 Multiplicative weights update method 299
24 Approximate tensor network contraction 305

PART III FAULT-TOLERANT QUANTUM COMPUTING 311
25 Basics of fault tolerance 314
26 Quantum error correction with the surface code 320
27 Logical gates with the surface code 325
Appendix **Background, conventions, and notation** 332
A.1 Quantum systems and bra-ket notation 333
A.2 The quantum circuit model 336
A.3 Noise in quantum gates and the NISQ era 338
A.4 Big-O notation 339
A.5 Complexity theory background 341

References 346
Index 415

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Preface

The first half of the twentieth century witnessed the foundation of three pillars of modern science: quantum mechanics, information theory, and computer science. In the latter half of the century, scientists began to connect these fields, first by exploring the implications of information itself being quantum—leading to the birth of quantum information theory. The nonclassical features of quantum information, such as no-cloning and entanglement, were identified as resources for novel applications, for instance, information theoretically secure communication. Alongside these developments were the observations of Benioff [120], Feynman [391], and Manin [737] that models of computation and simulation could be formulated within quantum mechanics—and that in some cases these models appeared exponentially challenging to simulate using a classical computer.

In 1985, Deutsch [346] further developed these early models of quantum computation and presented what was essentially the first quantum algorithm—a simple procedure that, with just one black-box query, could accomplish a task that classically requires two queries. Over the next decade, larger black-box separations were discovered, such as the Deutsch–Jozsa [347], Bernstein–Vazirani [129], and Simon's [940] algorithms, and finally, in 1994, the first truly *end-to-end* quantum algorithm was developed: Shor's algorithm [937] for factoring integers and computing discrete logarithms, bringing extensive ramifications for cryptography. This breakthrough demonstrated that quantum computers could not only speed up the solution of contrived black-box problems but, at least in theory, could provide faster solutions to important real-world problems. The discovery of Shor's algorithm transformed the field of quantum algorithms from a relatively niche topic into a major research area.

During the three decades since Shor's seminal discovery, the field of quantum algorithms matured significantly. For example, our knowledge of upper and lower bounds on the quantum query complexity of black-box

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

problems—often deduced through sophisticated, nonconstructive mathematical arguments—has been greatly expanded. Moreover, many additional quantum algorithms and subroutines—for example, primitives for quantum simulation and linear algebra—have been discovered, optimized, and subsequently generalized multiple times. Meanwhile, advances in hardware and the theory of fault-tolerant quantum computation have reached the point where it is conceivable that (some of) these algorithms might soon become implementable at scales large enough to surpass what can be done classically.

Nevertheless, the magnitude of available quantum speedups for real-world applications is often hard to assess and can be obscured by technical caveats, assumptions, and limitations in the underlying quantum algorithmic primitives. Despite being one of the oldest, Shor's algorithm for factoring arguably remains the cleanest example of a substantial quantum speedup with minimal caveats that targets a problem of significant real-world relevance. This survey aims to elucidate the true resource requirements of end-to-end quantum computing applications, and thereby aid in identifying the most likely applications for fault-tolerant quantum computers. Through this distinct perspective, the survey is intended to complement the wealth of existing quantum algorithms resources, including a number of review articles, lecture notes, textbooks, and the quantum algorithm zoo [586].

We highlight both the opportunities and challenges of currently known quantum algorithms. To truly understand the potential advantage of a quantum algorithm, it is necessary to consider its resource requirements in an end-to-end fashion. By this, we mean the cost of solving the full problem of interest to the user, not only the cost of running a given quantum circuit that is a subroutine of the full solution. One must consider all quantum and classical overheads: keeping track of classical precomputation and postprocessing, explicitly instantiating quantum oracles and data access structures, and ideally computing the constant prefactors of all quantum subroutines (including those overheads associated with fault-tolerant protocols and quantum error correction). We note, however, that this task is a major undertaking for complex quantum algorithms, and so has only been achieved for a minority of quantum algorithms in the literature. In addition to studying end-to-end quantum complexities, it is also necessary to compare any quantum results to the state-of-the-art classical solutions of the same problem, as well as known complexity-theoretic limitations.

We summarize the end-to-end complexities of several leading quantum application proposals (by which we mean quantum algorithms applied to a well-defined, real-world problem). The complexities of these applications are deduced from the complexities of their underlying primitives, which we review in detail. The modular structure of the book aids the high-level understanding

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

of the costs and tradeoffs coming from the various choices one makes when designing and compiling a quantum algorithm, as well as identifying the bottlenecks for a given application. On the technical front, this book does not attempt to advance the state of the art; rather, it aims to collect, synthesize, and contextualize key results in the literature. We consider algorithms in the quantum circuit model, which is arguably the best-studied model for quantum computation and renders the presented complexities hardware agnostic (although the overhead associated with executing these circuits in a fault-tolerant fashion can, of course, depend on details of the hardware). In order to obtain concrete bounds, we require oracles to be explicitly instantiated. We generally assume that quantum error correction of some form will be necessary, due to unavoidable imperfections inherent to all known quantum hardware modalities. As such, we typically consider the non-Clifford cost of quantum algorithms as the dominant cost, in keeping with leading quantum fault-tolerance schemes. Due to the general lack of application-scale experimental data, we focus on elucidating provable speedups, and we only mention noisy, intermediate-scale quantum (NISQ) algorithms in passing, where appropriate, since they are typically heuristic.

Throughout this book, we attempt to be thorough, but not exhaustive in presentation; we only aim to give a representative collection of references, rather than providing a complete list. Generally, we try to explain how asymptotic complexity statements arise from their underlying primitives, but technical results are typically presented without explicit derivation or proofs, for which we refer the reader to the cited references. Additionally, we often quote resource estimates from the literature without covering all of the application-specific optimizations to the underlying primitives that are required to arrive at the reported constant prefactors. We survey a number of quantum applications, primitives, and fault-tolerance schemes; however, the omission of other approaches does not indicate that they are unimportant. Also, the primary scope of this work excludes substantial topics, such as quantum sensing or communications, measurement-based quantum computing, adiabatic quantum computing and quantum annealing, analog quantum simulators, quantum-inspired ("dequantization") methods, and tensor network algorithms—comments on these topics are provided in instances where they are relevant to our primary discussion.

An overarching takeaway of this survey is that the current literature generally lacks fully end-to-end analyses for concrete quantum applications. Consequently, in several parts of this survey, a fully satisfactory end-to-end accounting is not achieved. In part, this is due to certain technical aspects of the relevant quantum algorithms being underexplored, and in some cases also due to a lack of specific details on how the output of the quantum algorithm will

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

integrate into concrete computational workflows for future quantum computing users. Quantum algorithms research often works upward from algorithmic primitives to identify computational tasks with maximal quantum speedups, but these may not align with the tasks most relevant to the user. On the other hand, potential users themselves may not yet know exactly how they would use a new capability to advance their high-level goals. Yet, we find ourselves at a point in the history of quantum computing at which it behooves us to fill in these details and adopt this end-to-end lens. As more end-to-end applications are found, and with small fault-tolerant quantum computers now on the horizon, we expect the story to continue to evolve—this survey provides a snapshot of the state of play in roughly mid-2024. While improved quantum algorithms and approaches to quantum error correction and fault tolerance are likely to be discovered, classical computers continue to grow in scale and speed, and classical algorithms are also constantly refined and developed, thereby moving the goalposts for end-to-end quantum speedups. We hope the reader will find this book a valuable guide for navigating this complex and dynamic landscape.

How to use this book

This book does not need to be read from cover to cover. Instead, it has a modular structure, which enables readers to directly explore the applications and primitives relevant for their use case. To the extent possible, each numbered chapter and section has been written in a self-contained fashion and can be read independently from the rest of the book. Readers looking for a quick introduction to (or refresher on) the common notation, conventions, and background concepts that underlie the technical exposition in the main text are advised to begin with the Appendix, where we provide information on quantum mechanics, bra-ket notation, quantum circuits (and quantum computing more generally), big-O notation, and complexity theory.

Readers of the link-enabled online version are encouraged to navigate the book using the hyperlinks embedded throughout, which connect interdependent material. Both print and online readers may also find utility in the index, which organizes mentions of the important topics discussed in the book, including computational tasks and problems, quantum algorithmic tools and primitives used to solve those tasks, and finally competing classical methods for those tasks (in addition to other miscellaneous topics).

Acknowledgments

We thank Joao Basso, J. Kyle Brubaker, Christopher Chamberland, Andrew Childs, Isabel Franco Garrido, Helmut G. Katzgraber, Eric M. Kessler, Robin Kothari, Péter Kutas, Yi-Kai Liu, Pavel Lougovski, Carl Miller, Oskar Painter,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Nicola Pancotti, Simone Severini, Sophia Simon, James D. Whitfield, and Xiaodi Wu for helpful comments and conversations on various aspects of this survey.

After the first version of this material was released online, we conducted a self-managed, nonanonymized peer-review process. We are grateful to colleagues who agreed to help us by reviewing a subset of the survey. At the beginning of each chapter, we acknowledge those who reviewed that chapter as part of this process, and we also list them alphabetically here: Dong An, Eric Anschuetz, Ryan Babbush, Matthew Campagna, Earl Campbell, Marco Cerezo, Andrew Cross, Zohreh Davoudi, Vedran Dunjko, Glen Evenbly, Di Fang, Joshua Goings, Johnnie Gray, Sander Gribling, Thomas Häner, Matthew Hastings, Samuel Jaques, Robin Kothari, Richard Kueng, Lin Lin, Daniel Malz, Ashley Milsted, Ashley Montanaro, John Preskill, Patrick Rall, Patrick Rebentrost, Rolando Somma, Nikitas Stamatopoulos, Damian Steiger, Yuan Su, Ewin Tang, Ronald de Wolf, Nobuyuki Yoshioka, and Xiao Yuan.

Finally, we acknowledge the staff from across the AWS Center for Quantum Computing that enabled this project, with special thanks to Mike Sadowitz and Wendy Yu for their legal support. We are also grateful to the Institute for Quantum Information and Matter at Caltech, which is an NSF Physics Frontier Center. We thank Harry Atwater, Fiona Harrison, Tom Rosenbaum, and David Tirrell at Caltech, and Nafea Bshara, Peter DeSantis, James Hamilton, Andy Jassy, Simone Severini, and Bill Vass at AWS, for their involvement and support of the research activities at the AWS Center for Quantum Computing.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

PART I

AREAS OF APPLICATION

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Overview

To provide benefit, quantum computers must solve computational problems where the solutions are simultaneously valuable to the user and also difficult to obtain classically. Simply developing a quantum algorithm with a theoretical quantum speedup is not sufficient to meet these criteria; we must directly compare the performance of classical and quantum algorithms for concrete problems of interest.

In this part, we survey a number of specific computational problems where quantum algorithms have been proposed, organized by application area. We present an overview of these algorithms through an end-to-end lens, noting clearly the actual end-to-end problem that is being solved and the dominant resource cost/complexity (derived from the algorithmic primitives that are being used), and emphasizing noteworthy caveats. We list known resource estimates for implementing these algorithms on fault-tolerant quantum computers (we also comment in passing on NISQ implementations), and we compare to classical complexities for the same problem, both in a practical and asymptotic sense. The list of applications presented is not exhaustive, but represents a broad spectrum of the most well-studied applications proposed in the literature.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

1

Condensed matter physics

Condensed matter physics constructs and studies the behavior of simplified models designed to capture the universal physics of material systems. Phenomena of interest include magnetism, phase transitions, superconductivity, frustrated systems, topological phases, and the interplay of thermalization and many-body localization in closed systems. While many seminal models can be studied analytically in certain limits (e.g., the 1D and 2D classical Ising model), a number of seemingly innocuous models have proven exceedingly difficult to solve. This has led to some models, such as the Fermi–Hubbard model, becoming a proving ground for classical numerical methods. While there has been significant progress in recent decades in understanding the physics of these models through numerical simulation, it is still a challenging problem for many models and parameter regimes. As observed by Feynman [391], quantum computers have a natural advantage over their classical counterparts for simulating the simple Hamiltonians studied in condensed matter physics. While Feynman's proposal was more focused on analog simulation, digital quantum simulation of condensed matter systems has evolved into a major research direction. In this chapter, we focus on models whose end-to-end complexities have been well studied in the literature: the Fermi–Hubbard model, spin models, and the Sachdev–Ye–Kitaev (SYK) model.

The authors are grateful to Ashley Montanaro and Nobuyuki Yoshioka for reviewing this chapter.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

1.1 Fermi–Hubbard model

Overview

The Fermi–Hubbard model was originally introduced as a simplified model of electrons in materials [546], closely related to the tight-binding model. It displays a wide range of behaviors including metallic, insulating, and antiferromagnetic phases. The model has more recently found applicability in studying high-temperature superconductivity. The 2D Fermi–Hubbard model has a complex phase diagram that appears to reproduce universal (rather than chemical-specific) features of the phase diagram of cuprate high-temperature superconductors.

General analytic solutions are not known beyond 1D chains or specific parameter regimes—see [59] for a discussion—which has motivated the use of numerical methods to understand the physics of the Fermi–Hubbard model. More recently, there has been increased interest in understanding the nonequilibrium properties of the model such as its behavior following a quench.

Based on the current estimates, quantum simulation of Fermi–Hubbard models requires considerably fewer resources than simulations of molecules or solving optimization problems. This makes the Fermi–Hubbard model a promising candidate for early demonstrations of quantum advantage.

A detailed case study on the Fermi–Hubbard model is presented in [101], including descriptions of the parameters to probe open scientific questions and estimates of the utility of these computational capabilities.

Actual end-to-end problem(s) solved

The Fermi–Hubbard Hamiltonian on $M/2$ lattice sites is given by

$$H = -t \sum_{\sigma\in\{\uparrow,\downarrow\}} \sum_{\langle i,j\rangle} (c^\dagger_{i\sigma} c_{j\sigma} + c^\dagger_{j\sigma} c_{i\sigma}) + U \sum_i n_{i\uparrow} n_{i\downarrow} \,, \tag{1.1}$$

where $c_{i\sigma}$ are fermionic operators and $n_{i\sigma} \equiv c^\dagger_{i\sigma} c_{i\sigma}$ is the number operator, with t denoting the strength of the kinetic term, U the onsite interaction strength, and $\langle i, j\rangle$ a sum over nearest-neighbor lattice sites, given a lattice geometry. It is also possible to consider longer-range hopping terms, the inclusion of site-dependent chemical potentials, or additional "orbitals" per site.

Quantum simulation provides insights into both equilibrium and nonequilibrium physics. With regards to equilibrium physics, the primary computational task is to resolve and probe the properties of the phase diagram of the Fermi–Hubbard model, as a function of lattice geometry, parameter values (t, U), doping (the expected number of fermions divided by the number of sites), and temperature. This is achieved by preparing the thermal state $\rho \propto \mathrm{e}^{-\beta H}$ (with

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$\beta = 1/k_B T$, where k_B is the Boltzmann constant and T the temperature) or at zero temperature the ground state $|E_0\rangle$ for the Fermi–Hubbard Hamiltonian instantiated by the given parameters, and measuring the expectation values of a set of physical observables to error ϵ. A thorough discussion of this end-to-end problem (at zero temperature) is provided in [1031], where it is shown how to perform the following steps:

- Prepare mean-field states in a given phase, for example, a BCS superconducting ground state.
- Adiabatically evolve from the mean-field Hamiltonian to the final Fermi–Hubbard Hamiltonian. The absence of a phase transition confirms the predicted phase.
- Measure observables, including density correlation functions $(n_{i\uparrow}+n_{i\downarrow})(n_{j\uparrow}+n_{j\downarrow})$, pair correlation functions $c^\dagger_{i\sigma} c^\dagger_{j\sigma'} c_{k\sigma'} c_{l\sigma}$, and dynamical correlation functions $\langle E_0|e^{iHt} A e^{-iHt} B|E_0\rangle$ (for operators A, B and ground state $|E_0\rangle$).

The difficulty of this problem depends on the parameter regime under consideration. The ground state in the weak coupling regime of $U < 4t$ is well understood, but questions remain in the intermediate ($4t \leq U \leq 6t$) and strong ($U > 6t$) regimes [847]. Challenges include precisely determining the phase boundaries and understanding the nature of the superconducting phase [399]. Progress has been made on this latter question in recent years, for example, by showing the absence of a superconducting phase at the physically relevant parameters of $U \sim 8t$ and 1/8th doping (see [847] for a more detailed discussion). Calculations are made challenging by small energy differences between competing phases, as well as the need to extrapolate from finite simulations to the thermodynamic limit.

The simulation of nonequilibrium quantum dynamics is of interest for modeling materials driven by an external field (e.g., an ultrafast laser pulse or an applied voltage), or following a quench in the Hamiltonian. Classically simulating nonequilibrium quantum dynamics has so far proven challenging and is a less well-studied problem than probing the equilibrium physics of the model. Example applications include modeling ultrafast spintronics (whereby lasers are used to manipulate spin degrees of freedom to control and store information) [1094], understanding photo-induced phase transitions [812], and clarifying the nature of thermalization in isolated quantum systems following a quench [837].

Dominant resource cost/complexity

Mapping the problem to qubits: Simulation of the Fermi–Hubbard model is most naturally performed in the second-quantized representation, as the regime

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

of interest is usually close to half-filling (for comparison, we refer to Chapter 2 on simulating molecules). The Jordan–Wigner mapping between fermions and qubits is typically used. Locality-preserving mappings have also been developed, which map fermionic operators to qubit operators acting on a constant number of qubits [1010, 344]. For an $L \times L$ lattice, we require $M = 2L^2$ qubits to simulate the spinful Fermi–Hubbard model using the Jordan–Wigner mapping.

Accessing the Hamiltonian: Quantum algorithms for simulating the Fermi–Hubbard model require access to the Hamiltonian. This is typically provided by block-encoding or Hamiltonian simulation.[1] The structure in the Fermi–Hubbard Hamiltonian reduces the costs of these subroutines. For example, performing a block-encoding using the linear combinations of unitaries (LCU) technique requires access to a PREPARE unitary and a SELECT unitary (we refer to Section 10.2 for definitions). The PREPARE unitary requires preparing a quantum state from classical data. Because the Fermi–Hubbard Hamiltonian has a small number of unique coefficients, the cost of this unitary can be reduced. Combining the results of [75, 1069, 225], one can implement an $(M(2t + U/8), O(\log(M)), \epsilon)$-block-encoding (see Eq. (10.1) for definition) of the Fermi–Hubbard Hamiltonian using

$$O\left(M + \log(M/\epsilon)\right)$$

non-Clifford gates.

As another example, the costs of product formula approaches for Hamiltonian simulation can exploit the fact that many terms in the Fermi–Hubbard Hamiltonian commute, due to their locality. We will explicitly discuss these costs below.

State preparation:

- Classical trial states: Approximate eigenstates obtained from a classical calculation can be prepared as quantum trial states; examples include Slater determinant states [1031], linear combinations of D Slater determinants (with complexity $\widetilde{O}(D)$ [394]–$O(MD)$ [998]), and matrix product states with bond dimension χ (with complexity $O(M\chi^2)$ [394]).
- Quantum trial states: Parameterized quantum circuits, in conjunction with variational quantum algorithms, have been proposed for preparing approxi-

[1] Hamiltonian simulation is used to explicitly simulate dynamics but can also be used implicitly to provide access to the Hamiltonian for use in static calculations, for example, in quantum phase estimation.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

mate energy eigenstates (see §NISQ implementations, below). Like classical trial states, the states prepared by these circuits can be used as inputs to other quantum algorithms that further refine the initial state, such as eigenstate filtering, or quantum phase estimation. Initial resource estimates for the Fermi–Hubbard model can be found in [215].

- Eigenstate preparation: There exist quantum algorithms that can prepare energy eigenstates using QSVT-based eigenstate filtering [688], where the cost scales as $1/\gamma$ with γ the overlap of the initial state with the desired eigenstate. Alternatively, adiabatic state preparation can be used, with a cost that depends on the gap between energy levels along the adiabatic path. Adiabatic state preparation was proposed as a method of classifying the phase diagram of the Fermi–Hubbard model [1031]. A discrete version of the adiabatic approach based on qubitization and quantum phase estimation (QPE) was numerically investigated in the context of preparing ground states of the Fermi–Hubbard model [673], and showed promising results for the small system sizes considered (see also [998]).
- Thermal states: A number of algorithms have been developed for preparation of thermal states, also known as Gibbs states. The most promising variants of these "Gibbs sampling" algorithms depend on the mixing time of a Markov chain (similar to classical Monte Carlo approaches for preparing Gibbs states), which is currently undetermined for the Fermi–Hubbard model.

Time evolution:

- As discussed above, Trotter approaches for Hamiltonian simulation can exploit beneficial features of the Fermi–Hubbard Hamiltonian, such as locality, fixed particle number, and commutativity of the terms [280, 300, 963]. For a Fermi–Hubbard model with η fermions on M lattice sites, pth-order Trotter methods can simulate time evolution for time τ up to error ϵ using

$$O\left(\frac{5^p M \eta^{1/p} \tau^{1+1/p}}{\epsilon^{1/p}}\right)$$

gates. Explicit gate counts for Trotterization can be obtained from [630, 300, 225, 909], which have focused on constant prefactors for low-order product formulas, rather than the asymptotic scaling.

Post-Trotter methods, such as [479], using quantum signal processing as a building block, can achieve similar scaling in M and τ. A suboptimal approach (i.e., not using the method of [479]), briefly discussed in [392], has

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

a gate complexity of approximately

$$44M^2(2t + 3U/8)\tau$$

T gates to simulate time evolution for time τ using quantum signal processing, neglecting logarithmic dependence on the error of the simulation.[2]

Measuring observables:

- Energies: QPE can be used to measure the energy eigenvalues of the Fermi–Hubbard Hamiltonian. Given access to (i) an initial state $|\psi\rangle$ that has sufficient overlap $\gamma = |\langle\psi|E_j\rangle|$ with the target eigenstate $|E_j\rangle$ and (ii) a unitary $U = f(H)$ that encodes the eigenspectrum of the Hamiltonian with a known, classically invertible relationship f, we can use QPE to project into the desired eigenstate and provide an estimate of the eigenphase ϕ_i of U—which can then be converted into an estimate of the eigenenergy of H using $\phi_i = f(E_i)$. QPE makes

$$O\left(\gamma^{-2}\|f'(H)\|^{-1}\epsilon^{-1}\log(\theta^{-1})\right)$$

calls to the unitary U encoding the spectrum of the Hamiltonian, where θ is the failure probability, and ϵ is the desired precision in the eigenenergy of H.[3]

A common choice for the unitary encoding the Hamiltonian is $U \approx e^{iHt}$ approximated via quantum algorithms for Hamiltonian simulation, where the approximation error must be balanced against the error from QPE. Using $U \approx e^{iHt}$ implemented via a second-order product formula results in a T gate count of $O(M^{3/2}/\Delta E^{3/2})$ to resolve the energy of the Fermi–Hubbard model to precision ΔE, neglecting the cost of initial state preparation and the dependence on the overlap and failure probability [630, 225]. Another common choice is to perform QPE on a quantum walk operator $W(H)$ which acts like $e^{i\arccos(H/\alpha)}$, where α is the normalization of the block-encoding of H. The operator $W(H)$ can be implemented exactly via qubitization [841, 139]. This results in a T gate scaling of $O(M^2/\Delta E)$, also neglecting the cost of initial state preparation and the dependence on the overlap and failure probability [75].

[2] Note that in [392], M is defined as the number of lattice sites, and so corresponds to $M/2$ here.
[3] It is possible to improve the complexity to $O(\gamma^{-1}\|f'(H)\|^{-1}\epsilon^{-1}\log(\theta^{-1}))$ using amplitude amplification if a sufficiently precise estimate of the eigenvalue is known, or to $O((\gamma^{-2}\Delta^{-1} + \epsilon^{-1})\|f'(H)\|^{-1}\log(\theta^{-1}))$ by exploiting knowledge of the gap Δ between the energy eigenstates to perform rejection sampling [139].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- Other observables: There have been few studies considering the costs of measuring observables other than the ground state energy using fault-tolerant quantum algorithms. In general, it is important to minimize the number of calls to the unitary U_ψ that prepares the desired state, as this is typically considered the dominant cost. Reference [1031] discussed methods for measuring density correlation functions $(n_{i\uparrow} + n_{i\downarrow})(n_{j\uparrow} + n_{j\downarrow})$, pair correlation functions $c^\dagger_{i\sigma} c^\dagger_{j\sigma'} c_{k\sigma'} c_{l\sigma}$, and dynamical correlation functions $\langle E_0 | e^{iHt} A e^{-iHt} B | E_0 \rangle$ (for operators A, B and ground state $|E_0\rangle$), including approaches for nondestructively measuring some of these observables. Some of these approaches can now be reframed as performing amplitude estimation [637] on U_O, a unitary block-encoding of the observable O with subnormalization factor α_O [853]. The measurement of similar observables using these modern algorithmic tools was studied in [9].

  A recent approach [549, 49] based on the quantum gradient estimation algorithm of [430] simultaneously computes the value of K (noncommuting) observables O_j. The algorithm makes $\widetilde{O}(K^{1/2}/\epsilon)$ calls to U_ψ and $U_\psi^\dagger$ (or $R_\psi = I - 2|\psi\rangle\langle\psi|$) and either $\widetilde{O}(K^{3/2}/\epsilon)$ calls to gates of the form e^{ixO_j} [549] or $\widetilde{O}(K/\epsilon)$ calls to a block-encoding of the observables [49]. The algorithm also requires $O(K \log(1/\epsilon))$ additional qubits. This approach has been considered in the context of measuring fermionic reduced density matrices and dynamic correlation functions [549].

Existing resource estimates

There have been a number of logical resource estimates for algorithms targeting both static and dynamic properties of the Fermi–Hubbard model. In Table 1.1, we present approximate resource estimates for simulations of the 2D 10×10 spinful Fermi–Hubbard model. The table presents the number of logical qubits and gates required to run the algorithm; these can be converted into physical resource estimates via methods for fault-tolerant quantum computation.

References [75, 1069] applied qubitization-based QPE to calculate the ground state energy to constant additive error. For a lattice with M spin sites, using the compilation of [75], the number of T gates scales as approximately (neglecting the dependence on the overlap and failure probability) [75, Eq. (61)]

$$\#T \propto \frac{(4t + U)M^2}{\Delta E},$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

and the number of logical qubits scales as approximately [75, Eq. (62)]

$$\#\text{Qubits} \sim M + \log\left(\frac{(2t + 0.5U)M^4}{\Delta E}\right).$$

References [630, 225] applied second-order Trotter-based QPE to calculate the ground state energy. In both references, rigorous but potentially loose upper bounds on the Trotter error are computed. For a lattice with M spin orbitals, using the compilation of [225], the number of T gates scales as approximately (neglecting the dependence on the overlap and failure probability) [225, Eqs. (C3), (D6), (D10), (E17), (F10)]

$$\#T \propto t\sqrt{t + U}\left(\frac{M}{\Delta E}\right)^{3/2},$$

and the number of logical qubits scales as approximately [225, Table II]

$$\#\text{Qubits} \sim (1 + \kappa)M,$$

where κ is a free parameter that controls the number of ancilla qubits used for a compilation technique known as Hamming weight phasing (which reduces the cost of applying identical arbitrary angle rotation gates in parallel) [421, 630], set to $\kappa = 0.25$ in [225] and in our Table 1.1.

Problem and method	# T gates	# Logical qubits	Parameters
Ground state energy via qubitized QPE [75, 1069]	$\sim 10^8$	~ 236	$U/t = 4$ and $\Delta E = 0.01t$
Ground state energy via Trotterized QPE [225, 630]	$\sim 5 \times 10^6$	~ 250	$U/t = 8$ and $\Delta E = 0.005E_{\text{tot}}$
Dynamics via fourth-order Trotter [392]	4.6×10^5	200	$T = 10/t$, $U = t$, and $\epsilon \leq 1\%$

Table 1.1 Logical resource estimates for quantum phase estimation (QPE) and dynamics simulation applied to a 2D 10×10 Fermi–Hubbard model. The QPE circuits target an energy error of ΔE. In the second row, E_{tot} denotes the ground state energy. The dynamics simulation runs for time T, and targets an error of less than 1% in a spatially averaged intensive observable, with Trotter errors bounded numerically via extrapolated small-scale simulations. The presented gate counts are for a single run of the circuit. For QPE, the number of required runs depends on the overlap between the initial state and the ground state (inverse polynomial dependence), as well as the desired failure probability of QPE. For dynamics simulations, the number of circuit repetitions depends on the precision to which one wants to estimate a given observable. The parameters for each problem vary between different rows of the table, and so cannot be directly compared (although the different methods for the same problem, e.g., ground state energy estimation, could be compared by adjusting the analyses in the original papers to the desired matching parameter values).

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

The methods described above for encoding the Hamiltonian spectra (qubitization and Trotter) can also be used to simulate the dynamics of the Fermi–Hubbard model. Trotter methods can be applied directly, while qubitization can be combined with quantum signal processing (QSP) to perform Hamiltonian simulation. In [392], a comparison was made between fourth-order Trotterization and qubitization+QSP for simulating time evolution of a 10×10 Fermi–Hubbard model. Trotter was determined to be the more efficient method, although this conclusion hinges on a Trotter decomposition with large steps (justified via numerical simulations). We note that the Trotter decompositions and analyses in [225, 392] are different, which hampers an immediate comparison. It may also be fruitful to compare with Hamiltonian simulation algorithms designed explicitly for simulating local Hamiltonians [479] (see discussion in [75]).

Caveats

In general, preparing the ground state of the Fermi–Hubbard model is known to be a hard problem, even for a quantum computer. This task has been proven QMA-hard for the Fermi–Hubbard model with a site-dependent magnetic field [910] and for the Fermi–Hubbard model with a site-dependent kinetic term strength (i.e., $t \to t_{ij}$ in Eq. (1.1)) [808]. While the complexity class of the canonical Fermi–Hubbard model is not yet known, when preparing the ground state via QPE or eigenstate filtering methods, it is necessary to prepare an initial state with an overlap that decays no worse than polynomially with system size; otherwise, the overall complexity will be superpolynomial. While numerical simulations on small system sizes have shown encouraging results [998, 673], it is still an open question as to whether this property holds for sufficiently large system sizes to enable extrapolation to the thermodynamic limit.

It is also important to note that this extrapolation of measured properties, computed at a range of finite system sizes, to the thermodynamic limit, has been observed to contribute a significant proportion of the uncertainty and errors in classical methods [666], and will also afflict quantum simulations.

Finally, it will be necessary to repeat simulations a large number of times. In order to measure a single observable to precision ϵ we require $O(1/\epsilon^2)$ incoherent repetitions of the simulation, or $O(1/\epsilon)$ using methods based on amplitude estimation. To map out and compute properties of the phase diagram or extract the phase following a quench, we may need to measure a large number of observables. In some cases, it may be necessary to re-prepare the initial state for each observable.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Comparable classical complexity and challenging instance sizes

The Fermi–Hubbard model has been a fertile environment for the development and testing of classical numerical methods for both static and dynamical properties. To the best of our knowledge, the largest exact diagonalization calculations performed to date are on systems with 17 fermions in 22 sites (44 spin sites), requiring 7.1 terabytes of memory [1063]. State-of-the-art approximate methods for computing the phase diagram include quantum Monte Carlo methods (variational QMC, determinantal QMC, diagrammatic MC, auxiliary-field QMC, diffusion MC), density matrix renormalization group (DMRG), coupled cluster methods, and impurity methods (dynamical mean-field theory, density matrix embedding theory), among others. These methods typically have an approximation parameter (e.g., the excitation degree in coupled cluster or the bond dimension in DMRG) which influences the scaling of the algorithm and the accuracy of the simulation. Modern numerical studies of the Fermi–Hubbard model typically cross-validate using a number of simulation methods [666, 900]. For example, [666] benchmarked a range of methods and performed sufficiently large and accurate simulations for extrapolation to the thermodynamic limit. That work concluded that "the ground-state properties of a substantial part of the Hubbard model phase space are now under numerical control," but that some uncertainties still remain for $4t \leq U \leq 8t$ and dopings near half-filling. For a recent review of numerical simulations of the Fermi–Hubbard model, we refer the reader to [847]. We also refer to [1054], which benchmarked a number of variational classical methods on a range of condensed matter systems, including the Fermi–Hubbard model.

The simulation of *dynamics* of the Fermi–Hubbard model appears to be more challenging for classical methods. For example, [326, 392] concluded that simulating the dynamics of a 10×10 lattice would be infeasible for tensor network techniques. Other classical approaches for simulating time evolution of the Fermi–Hubbard model include nonequilibrium extensions of dynamical mean-field theory [43] or Floquet methods [812].

Speedup

The speedup of quantum algorithms for computing static properties of the Fermi–Hubbard model, such as its ground state energy, is difficult to determine. In general, we know that closely related models are QMA-hard (see §Caveats, above) and so should be exponentially difficult for both classical and quantum computers. Assuming an initial state that has overlap with the target eigenstate that decays no faster than polynomially, then QPE can be used to efficiently measure the eigenenergy and project into the desired eigenstate. It does so with cost poly$(M/\Delta E)$, where the precise scaling depends on

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the quantum algorithm used. Exact classical methods such as exact diagonalization have a cost that scales exponentially with M or $1/\Delta E$. Approximate classical methods scale with an approximation parameter (e.g., bond dimension, number of excitations) which will depend on both M and ΔE. For example, [670, Fig. 4] shows the convergence of a tensor network calculation for the 2D Fermi–Hubbard model as a function of bond dimension and system size. For the small systems studied (up to 16×4 sites) the plots are consistent with the bond dimension scaling polynomially in $1/\Delta E$, with a weak dependence on the system size. If this holds for larger system sizes and across a range of system parameters, this would suggest that quantum algorithms provide only a polynomial speedup for computing the ground state energy.

Simulating the dynamics of the Fermi–Hubbard Hamiltonian requires polynomial resources using quantum algorithms, scaling almost linearly both in M and in the evolution time τ. By using carefully engineered interactions (e.g., deviating significantly from a square lattice) it can be shown that simulating the dynamics of the Fermi–Hubbard model on a planar graph is a BQP-complete problem and so is expected to be hard for classical computers in the worst case [93]. Supporting this observation, all known classical methods appear to scale exponentially in system size and simulation accuracy. For example, [392] used tensor network (matrix product state) approaches for simulating the dynamics of the Fermi–Hubbard model following a quench. When truncating the bond dimension to facilitate efficient classical simulation, they found that errors in the observables grew exponentially with time. While this supports the conclusion of an exponential quantum speedup, we note that classical approaches will likely continue to improve and be applied to increasingly large system sizes.

NISQ implementations

There have been a number of proposals (and experimental demonstrations) for simulating the Fermi–Hubbard model on NISQ hardware. Ground state calculations can be performed using the variational quantum eigensolver (VQE) [576, 871, 872, 218, 215], and experimental demonstrations have been carried out on lattices of size 1×8 and 2×4 using 16 superconducting qubits, yielding qualitative agreement with theoretical expectation [951].

Dynamics can be simulated using Hamiltonian simulation (typically Trotter methods) [300] and have been demonstrated for an 8×1 lattice on 16 superconducting qubits [64].

The simple Hamiltonian of the Fermi–Hubbard model makes it well suited to realization in analog quantum simulators, including ultracold atoms in optical lattices, trapped ions, and neutral atom arrays. It has been argued that some

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

local observables can be robust to errors in the simulation [836, 392], enabling analog simulations to already surpass classical methods for simulating dynamics. Nevertheless, it can be challenging to cool the analog fermionic system to its ground state. We refer the reader to [326, 461] for additional discussion on analog simulation.

Reference [300] considered an approach with flavors of both digital and analog simulation, moving to a cost model based on evolution time, rather than number of gates. This reduces the cost of simulating Trotterized dynamics of the Fermi–Hubbard model, compared to purely gate-based approaches.

Outlook

The Fermi–Hubbard model provides a long-standing and physically relevant computational challenge. The low gate counts and modest number of logical qubits required to compute ground state energies could make quantum algorithms competitive with leading classical approaches in challenging regimes. We note that further research is required to ascertain the costs for initial state preparation for these calculations. For the less well-studied task of simulating the dynamics of the Fermi–Hubbard model, quantum algorithms currently provide an exponential speedup over known classical algorithms. Nevertheless, as the Fermi–Hubbard Hamiltonian is sufficiently simple to be realized in many controlled physical systems, future fault-tolerant quantum computers will also have to compete against analog quantum simulators.

1.2 Spin models

Overview

Classical and quantum spin systems are prototypical models for a wide range of physical phenomena including magnetism, neuron activity, simplified models of materials and molecules, and networks. Studying the properties of spin Hamiltonians can also provide useful insights in quantum information science.

A number of scientific and industrial problems can be mapped onto finding the ground or thermal states of classical or quantum spin models, for example, solving combinatorial optimization problems, training energy-based models in machine learning, and simulating low-energy models of quantum chemistry [983].

Simulating the *dynamics* of quantum spin models is primarily of interest for quantum information science, as well as condensed matter physics or chemistry. For instance, such simulations are relevant for interpreting nuclear mag-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

netic resonance (NMR) [920, 806] or related spectroscopy experiments [273, 749].

Because of the natural mapping between spin-1/2 systems and qubits, as well as the fact that interactions in spin models are typically local, the resources required to simulate simple spin models using quantum algorithms can be much lower than for problems in areas like quantum chemistry or cryptography.

While our discussion will focus on quantum algorithms designed to be run on fault-tolerant quantum computers, the simple Hamiltonians of spin models are naturally realized in many physical systems. This has led to the use of analog simulators [157, 417], such as arrays of trapped ions or neutral atoms, for simulating the static and dynamic properties of interesting spin models. We will comment briefly on this below.

Actual end-to-end problem(s) solved

The most commonly studied spin models are those with pairwise interactions, referred to as 2-local Hamiltonians. We note that the interactions are not necessarily geometrically local, although this will be present in many models of physical systems. Given a graph $\mathcal{G}$ with N vertices $\{i\}$ and L edges $\{[i, j]\}$, we associate a classical or quantum spin with each vertex, and an interaction between spins with each edge. We can also add one-body interactions acting on individual spins. The Hamiltonian can then be written as

$$H = \sum_i \sum_{\alpha\in\{x,y,z\}} B_i^\alpha \sigma_\alpha^i + \sum_{[i,j]} \sum_{\alpha,\beta\in\{x,y,z\}} J_{ij}^{\alpha\beta} \sigma_\alpha^i \sigma_\beta^j , \tag{1.2}$$

where $\{\sigma_x^i, \sigma_y^i, \sigma_z^i\}$ denote the Pauli operators X_i, Y_i, Z_i acting on site i, and $\{B_i^\alpha\}, \{J_{ij}^{\alpha\beta}\}$ are coefficients. For classical spin Hamiltonians, the sums are restricted to Z operators. The Hamiltonian in Eq. (1.2) encompasses a wide range of spin models, including the following:

- The classical Ising model

$$H = \sum_i B_i Z_i + \sum_{ij} J_{ij} Z_i Z_j , \tag{1.3}$$

  which also describes the Hamiltonians arising from quadratic unconstrained binary optimization (QUBO) problems.
- The (quantum) transverse-field Ising model (TFIM)

$$H = B \sum_i X_i + J \sum_{ij} Z_i Z_j . \tag{1.4}$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- The Heisenberg model with a site-dependent magnetic field, defined in 1D with nearest-neighbor interactions by

$$H = \sum_j \left(B_j Z_j + J^x X_j X_{j+1} + J^y Y_j Y_{j+1} + J^z Z_j Z_{j+1} \right). \tag{1.5}$$

Across the different models, we can vary the dimension, locality of interactions (e.g., nearest-neighbor vs. fully connected vs. power-law), and values of the site-dependent coefficients in comparison to the interaction terms. The models can be extended beyond 2-local by considering couplings of 3 or more spins—see, for example, p-spin models, which are p-local [345]. The above definitions can be extended from spin-1/2 systems to higher-dimensional spin operators by generalizing the Pauli operators with their higher-dimensional counterparts.

For classical spin models, we seek to prepare the ground or thermal states of the model, as these may encode, for example, the solution to a combinatorial optimization problem, or a probability distribution that can be used for generative modeling. For quantum spin models, we similarly seek to compute ground or thermal states. However, because these are not classical states that can be easily extracted, we typically wish to sample observables with respect to these states. Examples include the energy, the magnetization of the system, and correlations between sites. In dynamics simulations of quantum systems, we seek to determine how observables of interest vary as a function of evolution time. Examples include the magnetization (used to infer the Hamiltonian in nuclear magnetic resonance [533] or related [172] experiments), or the growth of correlations between sites to probe thermalization. Digital Hamiltonian simulation can additionally extract certain quantities that may be hard to directly measure in experiments (e.g., time reversal) [402]. Since studies of quench dynamics often require preparation of simple states, such as product states or the ground states of classically solvable Hamiltonians, and the measurement of local observables, propagation under the Hamiltonian typically dominates the simulation cost. For lattice systems with N spins in D spatial dimensions, it is conventional to consider evolution times that scale as $\Omega(N^{1/D})$, as the system must evolve for at least this long in order for information to propagate across the system due to the Lieb–Robinson bound [261].

Reference [101] details a number of applications of simulating quantum spin models, with relevance to systems studied at Los Alamos National Laboratory, including the parameters of end-to-end simulations. Reference [374] presents an end-to-end assessment of using quantum computers to simulate quantum spin Hamiltonians for applications in NMR spectroscopy.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost/complexity

For a system of N spin-$\frac{1}{2}$ particles, we require N qubits to represent the state of the system. For N spin-S particles, the problem can be mapped to qubits in different ways, for example, using $N\lceil \log_2(2S+1) \rceil$ qubits [897] or using $2NS$ qubits [749].

Quantum algorithms for preparing the ground or Gibbs states of classical spin systems are discussed in detail in Chapter 4 (combinatorial optimization), and Section 9.2 (energy-based machine learning models), respectively. We will restrict our discussion to the resources required for performing time evolution of quantum spin models. The reason for this is that quantum algorithms for preparing ground or thermal states require similar primitives for Hamiltonian access to algorithms for time evolution (e.g., block-encodings or Hamiltonian simulation itself) and use these in conjunction with either (i) eigenstate filtering approaches [689, 688] based on quantum singular value transformation, (ii) adiabatic state preparation, (iii) quantum phase estimation from a trial state, or (iv) quantum algorithms for thermal state preparation, that is, Gibbs sampling. More detailed discussions of these algorithms and their caveats can be found in the corresponding sections, as well as in the discussion of quantum algorithms for simulating molecules and materials (Section 2.1) or the Fermi–Hubbard model (Section 1.1), where preparing (approximate) eigenstates is the primary topic of interest. All of these algorithms depend on either an overlap between the trial state and the target state, the minimum gap along an adiabatic path, or the mixing time of a Markov chain—all of which are difficult to bound in the general case.

When simulating the time evolution of spin systems via Hamiltonian simulation, the most efficient algorithms exploit the locality of interactions in the Hamiltonian, and the resulting commutation structure. For 2-local spin-1/2 systems on a D-dimensional lattice with nearest-neighbor geometric locality, algorithms with almost optimal gate complexity are known for performing time evolution. Reference [280] showed that the gate complexity of the $(2k)$th-order product formula scales as

$$O\left((Nt)^{1+1/2k}/\epsilon^{1/2k}\right)$$

to simulate time evolution for time t to accuracy ϵ, using a Hamiltonian given in the Pauli access model. Note that this expression suppresses the 5^{2k} constant prefactor present in $(2k)$th-order Trotter (see Eq. (11.1)). Similarly, [479] gave an algorithm with complexity $O(Nt \cdot \mathrm{polylog}(Nt/\epsilon))$ for Hamiltonians given in the sparse access model. In contrast, note that approaches that are asymptotically optimal in the black-box setting, such as quantum signal processing, have a gate complexity of $O(N^2 t + N \log(1/\epsilon))$ using a block-encoding based on lin-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ear combinations of unitaries (LCU)—the normalization of the block-encoding is $\alpha = O(N)$ and the gate complexity to implement the block-encoding is also $O(N)$.

Spin Hamiltonians with power-law interactions were studied in [993, 286], that is, where the interaction strength between spins i and j depends inversely on a power of the distance between the spins, denoted by $\|i-j\|_2$. (This implies that the model is translationally invariant.) For a D-dimensional lattice with 2-local interactions with interaction strengths scaling as $1/\|i-j\|_2^{\alpha}$, $(2k)$th-order Trotter gives a gate complexity scaling as (as above, suppressing the 5^{2k} constant prefactor present in $(2k)$th-order Trotter) [286]

$$\widetilde{O}\left(\begin{array}{ll} N^{3-\frac{\alpha}{D}(1+1/2k)+1/k} t^{1+1/2k} \epsilon^{-1/2k} & \text{if } 0 \leq \alpha < D, \\ N^{2+1/2k} t^{1+1/2k} \epsilon^{-1/2k} & \text{if } \alpha \geq D \end{array} \right).$$

Further improvements are possible in cases where the Hamiltonian coefficients are efficiently computable by an oracle, or if certain truncations can be performed [721]. Focusing on the $D = 1$ case, if one were to directly apply quantum signal processing based on a block-encoding via the linear combination of unitaries approach, the scaling of the gate complexity would be

$$\widetilde{O}\left(N^2 t + N \log(1/\epsilon)\right).$$

These asymptotic complexities are complemented by the constant prefactor analyses discussed in the following section.

For estimating expectation values of observables to precision ϵ, one can either consider directly sampling and then re-preparing the state of interest (scaling as $O(1/\epsilon^2)$), or coherent approaches based on amplitude estimation (scaling as $O(1/\epsilon)$, but requiring a longer coherent circuit depth). Measurements of simple observables, such as the magnetization, can be obtained through the computational basis measurements on single qubits. For more complicated observables, one can consider the approaches in [853, 549, 49], discussed in more detail in Chapter 2 (quantum chemistry).

Existing resource estimates

A number of logical resource estimates for simulating the dynamics of spin systems and for finding their ground states via quantum phase estimation have been reported in the literature. In such calculations, it is necessary to optimize the constant prefactor contributions from implementing the algorithmic primitives used. A detailed comparative study on simulating the dynamics of a 1D nearest-neighbor Heisenberg model (Eq. (1.5)) was reported in [283], comparing the logical qubit and T gate counts of product formulas, Taylor series, and

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

quantum signal processing. The two most efficient approaches are shown in the first two rows of Table 1.2.

On a fault-tolerant quantum computer, arbitrary angle rotation gates must be synthesized using a sequence of T and Clifford gates [625]. The number of T gates to synthesize a group of parallel rotation gates can be reduced if they share the same angle [421, 630, 225], a method known as Hamming weight phasing. This technique can be exploited in fault-tolerant compilations of algorithms simulating physical spin systems, which often exhibit features such as translational invariance.

In addition to the entries given in Table 1.2, fault-tolerant approaches to simulating NMR [806, 374] and muon spectroscopy [749] experiments, which are effectively spin model simulations, have been considered.

Caveats

When formulated as a decision problem, determining the ground state energy for 2-local classical and 2-local quantum spin models is NP-complete [94, 726] and QMA-complete [607], respectively. As such, we do not expect quantum algorithms to provide efficient solutions to these problems in the general case. Nevertheless, given the success of classical heuristics for these problems, one may hope to observe a similar benefit from quantum heuristic algorithms, such as Monte Carlo–style Gibbs sampling algorithms.

In contrast, simulating the dynamics of spin models is a BQP-complete problem [705]; it is likely one of the most simple beyond-classical calculations that could be performed on a future fault-tolerant quantum computer. While such a computation would be of great scientific interest, providing new insights in quantum information and many-body physics, it is currently unclear whether dynamics simulations of large systems will have a direct impact on industrially relevant applications.

Comparable classical complexity and challenging instance sizes

Exact classical simulations of quantum spin models are exponentially costly in system size. Exact simulations that consider a time evolution long enough for information to propagate across the system (as per the Lieb–Robinson bound) are limited to around 50 spins using the largest classical supercomputers [492, 283].

Approximate classical algorithms for studying quantum spin systems include tensor network approaches and quantum Monte Carlo (QMC) methods. These methods provide empirically accurate results for computing the ground states of physically motivated spin systems, in particular those with local interactions, in low dimensions. For example, the ground states of local, gapped 1D

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Problem	Method	# Spins	# T gates	# Logical qubits	Parameters
1D Heisenberg dyn.	QSP	50	2.4×10^9	67	$B_j \in [-1, 1], J^x = J^y = J^z = 1, t = N, \epsilon = 10^{-3}$ [283]
1D Heisenberg dyn.	Trotter (sixth order)	50	1.8×10^8	50	$B_j \in [-1, 1], J^x = J^y = J^z = 1, t = N, \epsilon = 10^{-3}$ [283]
2D NN TFIM[a] dyn.	Trotter (fourth order)	100	1.7×10^5	100	$t = 10/J, B = J, \epsilon = 10^{-2}$ [392, 147]
2D $1/r^2$ TFIM dyn.	Trotter (fourth order)	100	1.5×10^7	100	$t = 10/J, B = J, \epsilon = 10^{-2}$ [392]
2D Heisenberg ground state with nearest- and next-nearest-neighbor interactions	Qubitized QPE	100	10^8	112	$\epsilon = 10^{-2}, J_1 = 1, J_2 = 0.5, B_j = 0$ [1069]

Table 1.2 Logical resource estimates for quantum phase estimation (QPE) and dynamics simulation (dyn.) applied to different spin models. The presented gate counts are for a single run of the circuit. The results presented in rows 1 and 2 can be compared to each other, and both target an error of $\epsilon = 10^{-3}$ in the operator norm distance between the ideal and implemented time evolution unitary. While [283] presents both analytic and empirical Trotter error bounds, the gate count presented in the table is that resulting from the empirical bound, though we remark that more recent analytic bounds are close to matching the empirical bounds [286]. The results presented in rows 3 and 4 can be compared to each other, and determine the number of Trotter steps used empirically by targeting an error of $\epsilon = 10^{-2}$ in a particular spatially averaged local observable, and then extrapolating this behavior to larger system sizes.

[a] 2D nearest-neighbor transverse-field Ising model.

Hamiltonians have area law entanglement [504]; thus, they can be efficiently represented by matrix product states, a type of tensor network. The situation in higher dimensions is less clear, as an area law for gapped 2D local Hamiltonians has not been proven, and such a relation does not imply efficient representation via classical tensor network approaches such as projected entangled pair states (PEPS) [503]. Indeed, the 2D local Hamiltonian problem with the constraint that the ground state obeys area laws is QMA-complete [545]. Other approximate classical methods, including QMC approaches, can also be effective for preparing low energy states of spin systems. For example, [1054] provides benchmarks for both tensor network and QMC-based variational methods applied to spin systems in a range of lattice geometries.

In contrast, these methods are less accurate when performing simulations of quantum spin dynamics [912, 907]. In many of these systems, the entanglement entropy grows linearly with time [219], resulting in a cost that grows exponentially with time for tensor network approaches targeting fixed accuracy (see, e.g., [921, 501] for counterexamples). For example, it was claimed in [392] that simulations of the dynamics of the 2D TFIM for $N = 100$ spins would be far beyond the current capabilities of tensor network methods [392].

Many physical systems are subject to strong interactions with their environment, which limits their coherence times. In these cases, the behavior of the system can often be reproduced by simulating a smaller number of spins (e.g., $N \leq 30$) and accounting for the interactions with the environment through physically motivated heuristics [1045]. Such simulations (accessible via open-source software libraries) are used to analyze NMR [533] and muon spectroscopy experiments [172].

Speedup

The speedup for computing the ground states of quantum spin Hamiltonians over classical approximate methods (such as tensor networks or QMC) is currently an open research question; it depends on the complexity of being able to prepare good approximations of the ground state using quantum algorithms for cases where classical trial states are unable to efficiently and accurately represent the ground state [670].

The simulation of quantum spin dynamics appears to be exponentially costly using all known classical methods. As such, quantum algorithms for Hamiltonian simulation would provide an exponential speedup for this task. This would likely provide insights in quantum information and many-body physics. As an example, such systems could study the competition and interplay between thermalization and many-body localization in quantum systems.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

NISQ implementations

Quantum spin models are commonly used as benchmark systems for NISQ algorithms—for example, finding ground states [595], simulating dynamics [880], and probing thermalization [762]. For instance, variational algorithms have been applied to spin systems with up to 24 qubits, showing rapid convergence to the ground state as the number of layers of the variational circuit was increased [174, 604].

The Hamiltonians of spin models are also naturally realized in a wide range of physical systems, including trapped ions or neutral atoms [157, 417]. For example, recent experiments in neutral atom systems have studied the dynamics of on the order of 200 spins, which went beyond the capabilities of classical simulation via matrix product state approaches [371, 906]. Analog simulators are already an important tool providing new scientific insights, and they set a high bar for the future performance of fault-tolerant approaches to simulating spin systems. Nevertheless, analog simulators are restricted in the complexity of the models that they can simulate (e.g., it may be more challenging to study site-dependent impurity models or composite spin–fermion systems), and they are susceptible to errors from miscalibration and interactions with the environment.

Outlook

Simulating the behavior of spin systems is arguably one of the most natural tasks for quantum computers, while being exponentially costly using all known classical methods. Such simulations can provide important insights into questions in quantum information science and many-body physics. Spin system simulations are also relevant to condensed matter physics and chemistry, since spin systems can act as models for more complex systems in those fields.

Logical resource estimates for quantum algorithms simulating spin systems are among the lowest known for beyond-classical tasks. Nevertheless, analog quantum simulators are already able to natively simulate the dynamics of hundreds of spins. In order to surpass these capabilities, digital approaches may need to consider more complex observables or target better accuracies that are only achievable using devices capable of quantum error correction.

In addition, for many systems of scientific interest in related fields, such as chemistry or condensed matter physics, decoherence-inducing interactions with the environment often limit the required simulation sizes. Identifying applications where accurate dynamics simulation of large spin models is required would increase the impact and applicability of quantum algorithms in this area.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

1.3 SYK model

Overview

The Sachdev–Ye–Kitaev (SYK) model [889, 622] is a simplified model of a quantum black hole that is strongly coupled and "maximally chaotic," but still analytically tractable. This remarkable and, to date, unique combination of properties has led to great activity surrounding SYK. It has applications in high-energy physics through its connections to black holes and quantum gravity, and it has applications in condensed matter physics as a model of quantum chaos and scrambling, which sheds light on phases of matter in strongly coupled metals [882, 945]. While many interesting properties of the SYK model can be computed analytically in certain limits, not all properties qualify, and questions remain about the behavior of the model outside of these limits—these questions can potentially be addressed numerically by a quantum computer.

Actual end-to-end problem(s) solved

The SYK model has many variants; a common version to consider is the four-body ($q = 4$) N-site Majorana fermion Hamiltonian with Gaussian coefficients

$$H_{\mathrm{SYK}} = \frac{1}{4 \cdot 4!} \sum_{i,j,k,\ell=1}^{N} g_{ijk\ell}\, \chi_i \chi_j \chi_k \chi_\ell \,, \tag{1.6}$$

where χ_i denote Majorana fermion mode operators obeying the anticommutation relation $\chi_i \chi_j + \chi_j \chi_i = 2\delta_{ij} I$ (where I denotes the identity operator, and δ_{ij} the Kronecker delta symbol), and $g_{ijk\ell}$ are coefficients drawn independently at random from a Gaussian distribution with zero mean and variance $\sigma^2 = 3! g^2 / N^3$ (with g the tunable coupling strength). The normalization of Eq. (1.6) matches the convention of [78] and ensures that the ground state energy of H_{SYK} is extensive (i.e., growing linearly in N).

In the limit of a large number of local degrees of freedom $N \to \infty$ and at strong coupling $\beta g \gg 1$ (where β is the inverse of the temperature), analytic predictions can be computed for the asymptotic behavior of some properties. While these arguments are not mathematically rigorous, in practice they provide a consistent picture for the SYK model and provide insights into quantum gravity and quantum chaos. However, questions remain about the wealth of properties out of reach by taking limits or the nonasymptotic regime of parameters. For example, it has been challenging to rigorously calculate the density of states at arbitrary energies and the ground state energy in the large-N limit [316, 78, 511]. These problems can potentially be probed numerically on a quantum computer.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Generally speaking, this often reduces to performing the following task on the quantum computer: given as input an instance of H_{SYK} (generated by choosing the couplings $g_{ijk\ell}$ at random) and an observable O, estimate the expectation value $\mathrm{tr}(\rho O)$, where ρ could be, for instance, (i) the ground state of H_{SYK}, (ii) the thermal state $\rho \propto \mathrm{e}^{-\beta H_{\mathrm{SYK}}}$, or (iii) a time-evolved state $\rho = \mathrm{e}^{\mathrm{i}H_{\mathrm{SYK}}t}|0\rangle\langle 0|\mathrm{e}^{-\mathrm{i}H_{\mathrm{SYK}}t}$ from an easy-to-prepare initial state $|0\rangle$, among other possibilities. The observable O could be a local operator or even H_{SYK} itself. Another case is for O to be composed of t-dependent time-evolution unitaries $\mathrm{e}^{\mathrm{i}H_{\mathrm{SYK}}t}$.

For example, computing the ground state energy corresponds to taking ρ to be the ground state of H_{SYK} and O to be H_{SYK}, and computing a 4-point out-of-time-ordered correlation function corresponds to taking ρ to be the thermal state at inverse temperature β and O to be $A\mathrm{e}^{\mathrm{i}H_{\mathrm{SYK}}t}B\mathrm{e}^{-\mathrm{i}H_{\mathrm{SYK}}t}A\mathrm{e}^{\mathrm{i}H_{\mathrm{SYK}}t}B\mathrm{e}^{-\mathrm{i}H_{\mathrm{SYK}}t}$, where A and B are few-body operators [555]. In another example, [204, 799] give a detailed proposal to "simulate quantum gravity in the lab" via computing expectation values of observables and states formed via simulation of the SYK model.

Depending on the ultimate end-to-end goal, one may need to repeat this calculation for many different O or for many instances of H_{SYK}, for example, to compute an ensemble average.

Dominant resource cost/complexity

Mapping the problem to qubits: To simulate the SYK model on a quantum computer, the Majorana operators are represented by strings of Pauli operators according to the Jordan–Wigner representation (e.g., [407]). As a result, the Hamiltonian H_{SYK} on N Majoranas becomes a linear combination of multi-qubit Pauli operators over $N/2$ qubits. Methods for Hamiltonian simulation given a classical description of the Hamiltonian as a linear combination of Pauli strings (the Pauli access model) typically introduces into the complexity a dependency on the number of terms, N^4, and on the 1-norm of Pauli coefficients, denoted by λ, which for typical SYK instances is seen to be $\lambda = O(gN^{5/2})$ (see [78, Eq. (16)]).

State preparation: To solve the problem of estimating $\mathrm{tr}(\rho O)$, one must be able to prepare the $(N/2)$-qubit state ρ. In some cases, ρ could simply be a product state, which is trivial to prepare. If ρ is the thermal state at inverse temperature β, then algorithms for Gibbs sampling would be used to prepare the state. Due to the chaotic properties of SYK and the fact that the system is expected to thermalize quickly in nature, one expects that Monte Carlo–style Gibbs samplers (e.g., [984, 256, 939, 856, 260]) have a favorable poly(N) gate

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

complexity, but the exact performance is unknown. If ρ is the ground state of H_{SYK}, there are several methods for preparing ρ, including projection onto ρ by measuring (and postselecting) an ansatz state ϕ in the energy eigenbasis using quantum phase estimation (QPE), or by adiabatic state preparation. The cost of either of these methods depends on details such as which ansatz state is used (in particular, its overlap with ρ), the adiabatic path, and the spectrum of H_{SYK}—in both cases, in the absence of evidence to the contrary, the scaling can be exponential in N. In [511], a poly(N)-time quantum algorithm for preparing states ρ achieving a constant-factor approximation to the ground state energy of H_{SYK} was given, which could be used as ρ to probe low-energy properties of the system.

Time evolution: The calculation also requires simulating time evolution by H_{SYK}. This can be because O is a time-evolved operator, because the state ρ corresponds to a time-evolved state, or simply as a subroutine for QPE or Gibbs sampling, mentioned above. Reference [407] proposed a scheme for simulating time evolution using a first-order product formula approach to Hamiltonian simulation. That is, it implements the unitary $e^{iH_{\mathrm{SYK}}t}$ to precision ϵ, with gate complexity $O(N^{10}g^2t^2/\epsilon)$. However, this steep scaling with N suggests that accessing large system sizes will be difficult with this method. Reference [78] later gave a method with better N dependence, achieving gate complexity $O(N^{7/2}gt + N^{5/2}gt\,\mathrm{polylog}(N/\epsilon))$, leveraging qubitization with quantum signal processing. This gate complexity grows more slowly than the number of terms in H_{SYK} (i.e., $O(N^4)$), a feat that is only possible because the simulation method generates the SYK coupling coefficients pseudorandomly: to construct the block-encoding of H_{SYK}, they perform the PREPARE step in the linear combination of unitaries using a shallow quantum circuit composed of polylog(N) random two-qubit gates, producing a state for which the N^4 amplitudes are distributed approximately as independent Gaussians. Further reduction in the gate count would be bottlenecked by the 1-norm λ of the coefficients of H_{SYK}; however, note that [1059] suggests gravitational features may remain even if the Hamiltonian is substantially sparsified, which could reduce the number of terms and the value of λ.

Measuring observables: Finally, given the ability to prepare a purification of ρ and supposing O is unitary (if it is not, it could be decomposed into a sum of unitaries and each constituent computed separately), estimating the expectation value $\mathrm{tr}(\rho O)$ to precision ϵ can be done by overlap estimation, costing $O(1/\epsilon)$ calls to the routine that prepares ρ and to the routine that applies O. If the purification of ρ cannot be prepared, the cost is $O(1/\epsilon^2)$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Existing resource estimates

Reference [78] compiled the dominant contributions in their approach to Hamiltonian simulation into Clifford + T gates, and they found that at $N = 100$, implementing e^{iHt} requires fewer than $10^7 gt$ T gates, and at $N = 200$, it requires fewer than $10^8 gt$ T gates. Both of these figures are for a single circuit; that is, they do not include the cost of averaging over many SYK instances. The T-count can be turned into an estimate of the running time and number of physical qubits; see the discussion in Part III on fault-tolerant quantum computation.

Caveats

Existing resource estimates only focus on simulating the dynamics of SYK models, but the proposed classically challenging problems involve static properties such as density of states and properties of thermal states. Probing these static properties in an end-to-end fashion would likely require preparing thermal states, ground states, or other kinds of low-energy states, in addition to being able to implement e^{iHt}. The cost of preparing these states is unknown and difficult to assess analytically. Another caveat is that the gate counts quoted above do not take into account the $O(1/\epsilon)$ scaling of reading out an observable to precision ϵ, or any repetitions for different instances of H_{SYK} required for making inferences about the physics of SYK.

Comparable classical complexity and challenging instance sizes

As mentioned above, one of the reasons that the SYK model is appealing is that many properties can be computed analytically in certain limits. Other properties that would be of interest to numerically compute on a quantum computer require poorly scaling classical methods. Exact diagonalization of systems consisting of more than roughly 50 Majoranas would be challenging due to the exponential growth of the Hilbert space, which has dimension $2^{N/2}$. For example, [316] and [408] gave a variety of numerical results based on exact diagonalization up to $N = 34$ and $N = 36$, respectively.

Speedup

Hamiltonian simulation has poly(N) runtime, an exponential speedup over exact diagonalization, which is the go-to method for classical simulation of SYK-related problems. However, Hamiltonian simulation does not alone solve the same end-to-end problem as exact diagonalization; the persistence of the exponential speedup requires identifying specific interesting properties where the relevant initial states can also be prepared in poly(N) time, which is currently less clear.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

NISQ implementations

Experimental realizations of the SYK model have been proposed on several different experimental platforms [401, 851, 728]. However, even if these demonstrations can be realized, we do not expect this approach to scale in the absence of quantum error correction.

Outlook

Simulating time evolution of the SYK model on a quantum computer has relatively mild gate cost, due to the model's straightforward mapping to a qubit Hamiltonian. At the same time, it is difficult to simulate the SYK model on a classical computer, owing to its chaotic and strongly coupled nature. However, further work is needed to understand the entire end-to-end pipeline. It has not yet been identified which properties would be most valuable to compute on a quantum computer and how costly they will be. Computing these properties will likely involve far more than a single run of time evolution on a single instance of the SYK model, so the overall cost is likely to be much larger than what initial gate counts in the literature suggest.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

2

Quantum chemistry

Computational chemistry seeks to use computational methods to predict the physical properties and behaviors of atoms, molecules, and materials, and includes methods such as first-principles simulation, classical molecular dynamics, and cheminformatics. We will restrict our focus to first-principles simulations that treat chemical systems quantum mechanically (noting that these methods may be used within a larger workflow incorporating other techniques). Despite the apparent exponential cost of exact classical methods for this task, scientists have made incredible progress over the last century via increasingly sophisticated approximate methods. As a result, quantum chemistry is now a core part of several applications, including the analyses of chemistry experiments, the pharmaceutical drug discovery pipeline, and the optimization of materials for catalysts and batteries. Given the inherently quantum mechanical nature of these problems, it follows that several quantum algorithms have been proposed for computational chemistry [66]. In this chapter, we focus on simulating the electrons and vibrations of nuclei in molecules and materials. For further reviews of quantum computing for chemistry, we refer readers to [752, 229, 102, 783].

The authors are grateful to Ryan Babbush, Joshua Goings, and Ashley Montanaro for reviewing this chapter.

2.1 Simulating electrons in molecules and materials

Overview

We seek the energy eigenstates, thermal states (i.e., statistical ensembles of eigenstates at a given temperature), or dynamics corresponding to the Hamil-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tonian used to describe the electrons in molecules or material systems. The electrons interact with each other, in addition to fields produced by the nuclei and any external applied fields. In many systems it is appropriate to use the Born–Oppenheimer approximation, which treats the nuclei classically and fixes their spatial positions, separating the nuclear and electronic degrees of freedom.

Material systems can be described by a periodically repeating (i.e., translationally invariant) finite-size computational cell of interacting atoms. By simulating increasingly large computational cells and mitigating finite-size effects, we can extrapolate simulation results to the thermodynamic limit. This enables the measurement of bulk properties, such as magnetization, tensile strength, and thermal or electrical conductivity.

Actual end-to-end problem(s) solved

The Hamiltonian of a system consisting of K nuclei and η electrons interacting via the Coulomb interaction is (in atomic units)

$$H = -\sum_{i=1}^{\eta} \frac{(\nabla_i)^2}{2} - \sum_{I=1}^{K} \frac{(\nabla_I)^2}{2M_I} - \sum_{i,I} \frac{Z_I}{|r_i - R_I|} + \sum_{i \neq j} \frac{1}{2|r_i - r_j|} + \sum_{I \neq J} \frac{Z_I Z_J}{2|R_I - R_J|},$$

where ∇ is the gradient operator, r_i gives the position of the i-th electron, and R_I and Z_I give the position and charge of the I-th nucleus. This Hamiltonian can be discretized using a basis set $\{\phi_i(r)\}_{i=1}^{N}$ of electron spin orbital and $\{\chi_i(r)\}_{i=1}^{M}$ of nuclear orbital functions, or grid points, and can either be used with the time-dependent Schrödinger equation to simulate dynamics, or with the time-independent Schrödinger equation to obtain energy eigenstates. When simulating dynamics, it is necessary to use a basis set that is sufficiently flexible (or adaptive) to accurately describe the states at all times, as many chemical basis sets are highly optimized for ground state calculations and so are less suitable for dynamics calculations. It is often appropriate to make the Born–Oppenheimer approximation, fixing the positions of the nuclei, which are treated as classical point charges. The resulting electronic Hamiltonian at a fixed nuclear configuration is given by

$$H(\{R_I\}) = -\sum_{i} \frac{(\nabla_i)^2}{2} - \sum_{i,I} \frac{Z_I}{|r_i - R_I|} + \frac{1}{2} \sum_{i \neq j} \frac{1}{|r_i - r_j|} + V(\{R_I\}), \qquad (2.1)$$

where $V(\{R_I\})$ is the constant offset from the nuclear repulsion energy. In this case, the Hamiltonian is discretized using a basis set $\{\phi_i(r)\}_{i=1}^{N}$ of electron spin orbital functions or grid points. For many molecules at room temperature, the ground state of the electronic structure Hamiltonian is a good approximation

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

for the thermal state $\rho \propto e^{-\beta H}$ (with $\beta = 1/k_B T$, where k_B is the Boltzmann constant and T the temperature), as the electronic energy levels are well separated with respect to $k_B T$. This can be contrasted with the vibrational structure of molecules, where vibrational energies are on the order of $k_B T$, and so excited states are also populated at room temperature.

The electronic eigenstates (or thermal states) often provide a good starting description of a wide range of system properties, which then can be corrected by, for example, vibrational, rotational, relativistic (e.g., spin-orbit coupling) or entropic contributions. Preparing the desired electronic state for a given nuclear configuration is typically the first step in learning properties of the system. We then measure the expectation values of observables with respect to these states. Moreover, the electronic response to weak or slowly varying perturbations can often be described by a sequence of static calculations, for example, linear response theory for radiation absorption, or Born–Oppenheimer molecular dynamics [745] where one iteratively solves the electronic time-independent Schrödinger equation to obtain the forces on the nuclei, whose positions can then be updated using Newton's laws. Static calculations can be used to probe:

- Energy values (potentially across a range of nuclear configurations)—for electronic excitation energies at a fixed nuclear geometry, for determining molecular geometries by computing the electronic ground state energy at different geometries, and for finding reaction pathways and rates by computing energy differences between a sequence of geometries involved in a reaction. To obtain accurate predictions, the electronic energy values must be corrected by other contributions to the free energy.
- Determining transition probabilities between states—for reactions and optical properties.
- Differential changes in electronic energy in response to an applied field, for example, electronic or magnetic dipole moments, polarizability.
- Calculating forces on the nuclei, for use in molecular dynamics calculations—used in a range of applications, including protein folding and calculating drug molecule binding affinities.
- Orbital occupancies and correlation functions, as well as real- and imaginary-time Green's functions.

Properties of interest for materials include the following:

- Energy densities for given system parameters, to determine phase diagrams.
- Bulk properties, such as magnetization, thermal or electrical conductivity, and tensile strength.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- Particle densities and correlation functions between sites, as well as real- and imaginary-time Green's functions.

In order to understand how these observables vary as the system parameters (nuclear positions, atomic doping, temperature, applied field, etc.) are changed, the desired state may need to be prepared and measured a number of times.

Simulations of dynamics may be used to explicitly probe many of the equilibrium phenomena implicitly being probed above—for example, chemical reactions occurring at thermodynamic (quasi-)equilibrium—as well as additional nonperturbative or nonequilibrium phenomena that are difficult to implicitly describe as a sequence of static calculations, such as scattering from collisions, absorption of UV and X-ray radiation, charge-transfer dynamics, and optimal control. As a result, many of the same observables described above are still of interest and can be monitored as a function of time, including the following:

- Changes in kinetic or potential energy.
- Changes in particle densities or orbital occupancies.
- Changes in charge or spin densities.

Dominant resource cost/complexity

Mapping the problem to qubits: We discretize the electron positions using a basis of N spin orbital functions. For many basis sets, the discretization error decays as $1/N$ [488, 932] and is limited by the resolution of singularities in the Coulomb interaction from charge coalescence. A variety of functional forms have been considered for the electron orbitals (see Table 2.1 for a list of orbitals commonly considered in quantum computing). The optimal choice will be system dependent and must consider the following non-exhaustive list of factors:

- The resolution of the orbital, improved by matching the character of local vs. delocalized physics in the system to that of the orbital.
- The cost of computing the Hamiltonian, either in classical precomputation or (if required) coherently on a quantum device (see §Accessing the Hamiltonian, below).
- The properties of the resulting Hamiltonian (number of terms, norm, locality of terms, etc.) which determine the cost of accessing the Hamiltonian in algorithms.

The commonly used "Galerkin discretization scheme" discretizes the Hamiltonian via integrals over the basis functions, with one- and two-electron

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Representation	First-quantized	Second-quantized
Gaussians	[74][a] [416]	[1034]
Plane waves	[962]	[76]
Bloch/Wannier functions	Not yet studied	[561, 886]
Grids	[628]	[76, App. A]
Pseudo-spectral / Discrete variable representations	[601, 253, 287]	[76, 721]

Table 2.1 Representative references (chosen based on their discussion of their choice of representation) showing the use of different basis functions in quantum algorithms for the electronic structure problem.

[a] This reference is not technically a first-quantized representation, as antisymmetry is stored in the operators rather than the wavefunction, but it stores states in an analogously compressed way to first-quantized representations.

integrals

$$h_{ij} = \int \mathrm{d}r\, \phi_i^*(r) \left(-\frac{(\nabla)^2}{2} - \sum_I \frac{Z_I}{|r - R_I|} \right) \phi_j(r)$$
$$h_{ijkl} = \int \mathrm{d}r_1 \mathrm{d}r_2\, \frac{\phi_i^*(r_1)\phi_j^*(r_2)\phi_k(r_2)\phi_l(r_1)}{|r_1 - r_2|}\,. \tag{2.2}$$

Hamiltonians defined using grids, pseudo-spectral representations, or discrete-variable representations are not obtained from integrals over basis functions, as specified in Eq. (2.2), and the values of h_{ij}, h_{ijkl} are instead defined using finite difference formulas, and/or by their values at discrete grid points. An attractive feature of the Galerkin discretization scheme is that the discretization error is strictly positive. We refer readers to [287, 76, 244] for a more complete discussion.

We can represent electronic states on a quantum computer using either first or second quantized representations.

- For η electrons in N spin orbitals, first quantization uses η registers, which each contain $\log_2(N)$ qubits; each register enumerates which orbital its corresponding electron is in, and the wavefunction must then be anti-symmetrized to respect fermionic constraints [139]. The Hamiltonian of Eq. (2.1) in first quantization can be written as

$$H = \sum_{\alpha=1}^{\eta} \sum_{i,j=1}^{N} h_{ij} |i\rangle\langle j|_\alpha + \frac{1}{2} \sum_{\alpha \neq \beta} \sum_{i,j,k,l=1}^{N} h_{ijkl} |i\rangle\langle l|_\alpha \otimes |j\rangle\langle k|_\beta\,,$$

  where α, β index which of the electron registers the operators act on.
- In second quantization, antisymmetry is stored in the operators, which obey fermionic anticommutation relations: $\{a_i, a_j^\dagger\} = \delta_{ij} I$ and $\{a_i, a_j\} = \{a_i^\dagger, a_j^\dagger\} =$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

0 (where δ_{ij} denotes the Kronecker delta symbol, I denotes the identity operator, and $\{u, v\} = uv + vu$). The Hamiltonian of Eq. (2.1) in second quantization can be written as

$$H = \sum_{i,j=1}^{N} h_{ij} a_i^\dagger a_j + \frac{1}{2} \sum_{i,j,k,l=1}^{N} h_{ijkl} a_i^\dagger a_j^\dagger a_k a_l.$$

A number of fermion-to-qubit mappings have been studied; see [752] for discussion. Under the commonly used Jordan–Wigner mapping we require N qubits, where each qubit stores the occupancy of the corresponding spin orbital. These mappings induce a mapping of the Hamiltonian (and other observables) to qubit operators.

Accessing the Hamiltonian: Quantum algorithms for both static and dynamic simulations require access to the Hamiltonian. This is typically provided by block-encoding or Hamiltonian simulation.[1] A common block-encoding strategy for chemistry Hamiltonians is the linear combinations of unitaries (LCU) block-encoding, whereby the Hamiltonian is expressed as a linear combination of unitary operators $\sum_{i=0}^{L-1} c_i U_i$ (e.g., U_i could be products of Pauli matrices), and the block-encoding is then realized using the oracles PREPARE and SELECT[2] that act on the main register and a $\lceil \log_2(L) \rceil$ ancilla system as

$$\text{PREPARE}|0^{\lceil \log_2(L) \rceil}\rangle = \frac{1}{\sqrt{\lambda}} \sum_{j=0}^{L-1} \sqrt{|c_j|} |j\rangle$$

$$\text{SELECT} = \sum_{j=0}^{2^{\lceil \log_2(L) \rceil}-1} |j\rangle\langle j| \otimes \text{sign}(c_j) U_j,$$

where $\lambda = \sum_{i=0}^{L-1} |c_i|$. Then the sequence $\text{PREPARE}^\dagger \cdot \text{SELECT} \cdot \text{PREPARE}$ is a $(\lambda, \lceil \log_2(L) \rceil, 0)$-block-encoding of the Hamiltonian written as an LCU. The oracle SELECT can be implemented using the unary iteration method [75] or the approach of [1011]. The oracle PREPARE can be implemented by coherently loading coefficients stored in memory [75, 722, 140] or by computing coefficients on-the-fly using quantum arithmetic [601, 74, 73, 962]. In many cases, loading the coefficients for PREPARE is the bottleneck, with cost—in terms of the number of non-Clifford gates—scaling linearly with $\widetilde{L}$, the number of unique coefficients in the Hamiltonian (this scaling can be reduced to

[1] Hamiltonian simulation is used to explicitly simulate dynamics, but can also be used implicitly to provide access to the Hamiltonian for use in static calculations, for example, in quantum phase estimation.

[2] To be precise for $j \notin \{0, 1, \ldots, L-1\}$ we define $\text{sign}(c_j) U_j := I$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$O(\widetilde{L}^{1/2})$ using $O(\widetilde{L}^{1/2})$ ancilla qubits [722]). As a result, a number of algorithms have reduced the quantity of data to load by using compressed representations of the Coulomb Hamiltonian, achieved through tensor factorizations [140, 212, 669, 886] (see [712] for a recent unifying perspective on these approaches).

State preparation: Simulating the behavior of electrons in molecules and materials reduces to the task of preparing a desired state and measuring observables. The state to be prepared is typically an energy eigenstate, a thermal state, or a time-evolved state.

- Energy eigenstates: In the following discussion, we refer to the overlap $\gamma = |\langle\psi|E_j\rangle|$ between a desired eigenstate $|E_j\rangle$ and a given initial state $|\psi\rangle$, and the minimum gap Δ between the desired energy eigenvalue and other energy eigenvalues. Below, we list several methods for preparing energy eigenstates, or approximations to them.
  - Classical trial states: Approximate eigenstates obtained from a classical calculation can be prepared as quantum trial states, including Slater determinant states [1031], linear combinations of D Slater determinants (with complexity $\widetilde{O}(D)$ [394]–$O(ND)$ [998]), and matrix product states (MPSs) with bond dimension χ (with complexity $O(N\chi^2)$ [394, 144]). In [144] it was observed that MPS with modest bond dimension could have large overlaps with chemical systems of interest. Several of these methods have been adapted to simulations performed in first quantization [81, 551].
  - Quantum trial states: Parameterized quantum circuits, in conjunction with variational quantum algorithms, have been proposed for preparing approximate energy eigenstates (see §NISQ implementations, below). Like classical trial states, the states prepared by these circuits can be used as inputs to other quantum algorithms that further refine the initial state, such as eigenstate filtering, or quantum phase estimation. Initial resource estimates for models of materials systems can be found in [301].
  - Eigenstate filtering: Methods such as those in [689, 688] filter out undesired eigenstates using spectral window functions applied via quantum singular value transformation (QSVT) to a block-encoding of the Hamiltonian. The complexity to prepare the ground state (to infidelity ϵ, with failure probability less than θ) using this approach scales as

$$\widetilde{O}\left(\frac{\alpha}{\gamma\Delta}\log\left(\theta^{-1}\epsilon^{-1}\right)\right)$$

    calls to an $(\alpha, m, 0)$-block-encoding of the Hamiltonian, where $\alpha \geq \|H\|$ is a normalization factor of the block-encoding. For comparison to related

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

methods, we refer the reader to [688, 412]. These algorithms can also be adapted for the case where access to the Hamiltonian is provided by Hamiltonian simulation [358].

– Adiabatic state preparation (ASP): ASP can be used to prepare a target eigenstate (typically the ground state) by evolving from the corresponding easy-to-prepare eigenstate of an initial Hamiltonian $H(0)$ to the full Hamiltonian $H(1)$. Time evolution can be implemented using algorithms for Hamiltonian simulation. The total evolution time is typically chosen according to the heuristic

$$T \gg \max_{0 \le s \le 1} \frac{\|\frac{dH}{ds}\|}{\Delta(s)^2},$$

where s describes the adiabatic path $H(s)$ and $\Delta(s)$ is the spectral gap of $H(s)$. It is difficult to analytically bound this (highly system-dependent) complexity for molecular systems (see, e.g., [870]) motivating numerical studies on small molecules [1006, 644, 670, 966].

– Quantum phase estimation (QPE): The above techniques all provide methods of preparing approximate eigenstates, in some cases using promises on the gap Δ, or by exploiting pre-existing knowledge of the energy eigenvalue. Given an approximate eigenstate, and a unitary $U = f(H)$ that encodes the eigenspectrum of the Hamiltonian (with a known, classically invertible relationship f), we can use QPE to project into the desired eigenstate and provide an estimate of the eigenphase ϕ_i of U, which can then be converted into an estimate of the eigenenergy of H using $\phi_i = f(E_i)$. QPE makes

$$O\left(\gamma^{-2}\|f'(H)\|^{-1}\epsilon^{-1}\log\left(\theta^{-1}\right)\right)$$

calls to the unitary $U(H)$ encoding the spectrum of the Hamiltonian, where $\gamma = |\langle\psi|E_j\rangle|$ is the overlap between the state $|\psi\rangle$ input to QPE and the desired energy eigenstate $|E_j\rangle$, θ is the failure probability, and ϵ is the desired precision in the eigenenergy of H. It is possible to improve the complexity to

$$O\left(\gamma^{-1}\|f'(H)\|^{-1}\epsilon^{-1}\log\left(\theta^{-1}\right)\right)$$

using amplitude amplification if a sufficiently precise estimate of the eigenvalue is known, or to

$$O\left(\left(\gamma^{-2}\Delta^{-1}+\epsilon^{-1}\right)\|f'(H)\|^{-1}\log\left(\theta^{-1}\right)\right)$$

by exploiting knowledge of the gap Δ between the energy eigenstates to perform rejection sampling [139]. The unitary encoding the Hamil-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tonian is often chosen to be $U(H) \approx e^{iHt}$ approximated via quantum algorithms for Hamiltonian simulation. In this case, the approximation error in U must be balanced against the error from QPE. Alternatively, it is common to encode the Hamiltonian using a quantum walk operator $W(H)$ which acts like $e^{i \arccos(H/\alpha)}$ and can be implemented exactly via qubitization [841, 139]. The costs to implement $U(H)$ are inherited from the method used, based on the properties (e.g., commutativity, locality, number of terms, norm, cost of coherently calculating coefficients) of the Hamiltonian in the chosen spin orbital basis and representation. As indicated by the complexities presented above, QPE incurs an overhead from imperfect overlap with the target eigenstate, as well as from needing to suppress the failure probability of the method. We refer to Chapter 13 on QPE for a more detailed discussion of the latter issue, which can either be mitigated by repeating the calculation and using methods for probability amplification [792, 222], or by using window functions [143, 829, 456] to guarantee a desired confidence interval. The latter strategy appears to require fewer resources in practice, especially when considering the imperfect overlap with the target state [144].

- Thermal states: Several quantum algorithms have been proposed for preparing thermal states, also known as Gibbs states. The most efficient algorithms proceed by simulating the dissipative open system dynamics, and typically make repeated calls to a block-encoding of the Hamiltonian. The complexity of these methods for concrete electronic structure problems of interest has not yet been determined and depends on the spectral gap of the Lindbladian considered, at the desired temperature. Thermal states could be used as an approximation to the ground state, by choosing the temperature to be sufficiently low compared to the gap between the ground and first excited state [260]. Quantum algorithms for simulating open systems dynamics can also be adapted to directly prepare approximations to the ground state [351].
- Time-evolved states: A time-evolved state can be prepared using Hamiltonian simulation algorithms, which approximate the propagator to error ϵ (which provides an upper bound on the error in the resulting state). The cost of Hamiltonian simulation depends both on the algorithm used and the details of the Hamiltonian being simulated. Plane wave, grid, and pseudo-spectral (DVR) basis sets are well suited to simulations of dynamics, as they treat all points in space on an equal footing. For both the plane wave basis and pseudo-spectral DVR in first quantization, the scaling is [77, 962]

$$\widetilde{O}\left(\eta^{8/3} N^{1/3} t + \eta \log(1/\epsilon)\right)$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

using Hamiltonian simulation in the interaction picture [718], or

$$\widetilde{O}\left((\eta^{8/3}N^{1/3} + \eta^{4/3}N^{2/3})t + \eta\log(1/\epsilon)\right)$$

using qubitization with quantum signal processing [77, 962]. In the pseudospectral DVR, the cost scales as

$$\widetilde{O}\left((\eta^{7/3}N^{1/3} + \eta^{4/3}N^{2/3})\frac{t^{1+o(1)}N^{o(1)}}{\epsilon^{o(1)}}\right)$$

using high-order product formulas [81].

Measuring observables: Many proposed algorithms consider the ground or excited state energy of the chemical system as the observable of interest. This can be calculated directly using QPE, as discussed above. QPE (and related methods) can also be adapted for other application, for example, calculating absorption spectra of molecules [395].

In other applications, it may be necessary to measure observables other than the energy. In a fault-tolerant computation, it can be preferable to measure these observables through phase-estimation-like approaches, rather than direct measurement averaging, as the former is asymptotically more efficient and can be made robust to logical errors through repetition and probability amplification. Measurement schemes have been developed which achieve this using overlap estimation [637] (see Section 14.2 on amplitude estimation) or the approach of [549, 49] based on the quantum gradient estimation algorithm of [430]. Both approaches require access to a state preparation unitary U_ψ, and its inverse.[3] The algorithm based on overlap estimation can be formulated as performing amplitude estimation on U_O, a unitary block-encoding of the observable O with subnormalization factor α_O. The complexity to compute the expectation value to precision ϵ is $O(\alpha_O/\epsilon)$ calls to U_O and U_ψ (or the reflection $R_\psi = I - 2|\psi\rangle\langle\psi|$) and their inverses. This approach has been considered in the context of measuring correlation functions, density of states, and linear response properties (all in [853]), as well as energy gradients with respect to various parameters, which can be used to compute forces or dipole moments, and for which a range of estimation strategies are possible [805, 956].

The gradient-based algorithm simultaneously computes the value of M potentially noncommuting observables O_j by making $\widetilde{O}(M^{1/2}/\epsilon)$ calls to $U_\psi, U_\psi^\dagger$ (or R_ψ) and either $\widetilde{O}(M^{3/2}/\epsilon)$ calls to gates of the form e^{ixO_j} [549] or $\widetilde{O}(M/\epsilon)$ calls to a block-encoding of the observables [49]. The algorithm also requires

[3] Note that it can be substantially cheaper to directly execute the reflection $R_\psi = I - 2|\psi\rangle\langle\psi|$ used in both methods, rather than through the use of U_ψ, as the complexity of R_ψ does not depend on the overlap γ that appears in state preparation—see [688] for additional discussion.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$O(M \log(1/\epsilon))$ additional qubits. This approach has been considered in the context of measuring nuclear forces [805], fermionic reduced density matrices [549], and dynamic correlation functions [549].

Existing resource estimates

There are a large number of resource estimates for performing phase estimation to learn the ground state energies of molecular or material systems, which we list in Table 2.2 and Table 2.3. These resource estimates use compilation methods described in Part III on fault-tolerant quantum computing. We also note the existence of a software package that provides features for calculating the non-Clifford costs of QPE for the electronic structure problem [238].

There have been comparatively few studies of the logical resources required for the simulation of chemical dynamics. Recent work has computed the resources required to calculate the energy loss of charged particles moving through a medium ("stopping power"), as pertaining to nuclear fusion experiments [887]. End-to-end resource estimates were determined, including the costs of initial state preparation, measurement of observables, and repetitions across a range of parameters. The resource estimates for the end-to-end task ranged from roughly 2000 logical qubits and order-10^{13} Toffoli gates to roughly 30,000 logical qubits and order-10^{17} Toffoli gates.

Caveats

Existing resource estimates typically consider only a single run of phase estimation and assume that we have access to the desired energy eigenstate. As outlined above, both phase estimation and eigenstate filtering scale as $\Omega(\gamma^{-1}\Delta^{-1})$ when we have a lower bound on the gap. The "orthogonality catastrophe" suggests that the overlap of simple trial states with the desired eigenstate will decay exponentially as a function of system size. Although simulations will always be performed on finite-size systems, it is an open question as to when asymptotic behavior becomes problematic and whether initial states with sufficient overlaps can be prepared for systems of interest [998, 670, 144]. This issue may become more pressing for materials systems as we scale to the thermodynamic limit. In general, we know that the problem of finding the ground state of electronic structure Hamiltonians is QMA-hard [1035], but it is not yet known if these complexity-theoretic statements provide intuition for physically realistic Hamiltonians.

As noted above, to accurately resolve the system, a large basis set must be used, as the discretization error decays as $1/N$ where N is the number of spin orbitals considered. In practice, one typically repeats the calculation using increasingly accurate basis sets and then extrapolates to the continuum limit.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B881998C0DA5A2CC99794A8C39

Molecule(s) & references	Algorithm	Logical qubits	T/Toffoli gates per sample	Number of samples
FeMo-co (nitrogen fixation) [870, 140, 212, 669, 1013, 238]	2nd Q, THC qubitization, Gaussians 2nd Q, randomized compilation, Gaussian	2196 [669] ~193 [1013]	3.2×10^{10} [669] $\sim 3 \times 10^{12}$ [1013]	7 ~600
Cytochrome P450 (biological drug metabolizing enzyme) [437]	2nd Q, THC qubitzation, Gaussians	1434	7.8×10^{9}	7
Lithium-ion battery molecules [617, 962]	2nd Q, DF qubitization, Gaussians 1st Q, qubitization, plane waves	10^4–10^5 [617] 2000–3000 [962]	10^{12}–10^{14} [617] 10^{11}–10^{12} [962]	7 7
Chromium dimer [375]	2nd Q, sparse qubitization, Gaussians	~1300	$\sim 10^{10}$	7
Ruthenium catalyst (CO_2 fixation) [212]	2nd Q, DF qubitization, Gaussians	~4000	$\sim 3 \times 10^{10}$	7
Ibrutinib (drug molecule) [158]	2nd Q, Sparse qubitization, Gaussians	2207	1.1×10^{10}	7
Molybdenum catalysts (nitrogen fixation) [113]	2nd Q, DF qubitization, Gaussians	1000–8000	10^{11}–10^{12}	3–9[a]
Amyloid beta binding site fragment (Alzheimer's disease) [818]	2nd Q, DF qubitization, Gaussians	5000	10^{14}	7
Fullerene-encapsulated cyclic ozone (rocket fuel) [1029]	2nd Q, DF qubitization, Gaussians 2nd Q, sparse qubitization, plane wave dual	10^3 10^4	10^{12}–10^{13} 10^{15}	7

Table 2.2 Logical resource estimates for quantum phase estimation (QPE) applied to a range of molecular systems to compute a single energy eigenvalue. We list state-of-the-art resource estimates and refer to the references therein for prior estimates. The presented gate counts are for a single run of the phase estimation circuit. The number of samples is determined by the desired maximum failure probability, taken here as 0.1. In most cases, the number of samples is calculated using [792, Lemma 1] which assumes that the median value is taken from a number of incoherent repetitions of QPE. The success probability of a single run of QPE is $8/\pi^2$ [186]. QPE must be run a number of times if the overlap $\gamma \leq 1$, in general contributing a multiplicative cost of $O(1/\gamma^2)$ (though this may be reduced, as described in the main text). "Algorithm" denotes the phase estimation unitary considered (e.g., Trotterization, Qubitization) as well as details about the quantization scheme (1st or 2nd), basis used, or factorization method used to compile the unitary (sparse [140], SF = single factorized [140], DF = double factorized [212], THC = tensor hypercontraction [669]). The resource estimates presented can be for different numbers of electrons and orbitals, accuracies, and can have differing classical simulation complexities. As such, the results may not be directly comparable, even within a single row of the table.

[a] This resource estimate assumed an overlap of $\gamma < 1$, and a lower failure probability of 0.01.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B881998BC0DA5A2CC99794A8C39

Material(s) & references	Algorithm	Logical qubits	T/Toffoli gates per sample
Electron gas (prototypical model) [75, 630, 753, 962]	1st Q, qubitization, plane waves 2nd Q, qubitization [75]/Trotter [753], plane wave (dual)	1500–5000 [962] 100–1000 [75, 753]	10^9–10^{14} [962] 10^8–10^{11} [75, 753]
Lithium-ion battery materials [341, 934, 886, 142]	2nd Q, sparse/SF/DF/THC qubitization, Bloch orbitals 1st Q, qubitization, plane waves (w. pseudopotential)	10^5–10^6 [886] ~1000 [142]	10^{12}–10^{14} [886] ~10^{14} [142]
Transition metal catalysts nickel/palladium oxide [561, 416]	2nd Q, sparse qubitization, Bloch/Wannier orbitals 1st Q, sparse qubitization, plane wave (dual)	10^4–10^5 [561] 17,505 [416]	10^{10}–10^{13} [561] 10^{15} [416]
Magnesium/niobium alloys (corrosion resistant) [800]	2nd Q, qubitization, plane wave (dual)	9000–500,000	10^{14}–10^{19}
Nitrogen vacancy center in diamond (quantum sensing) [562]	2nd Q, DF qubitization, plane wave (projector augmented-wave method)	5000–150,000	10^{12}–10^{14}

Table 2.3 Logical resource estimates for quantum phase estimation (QPE) applied to a range of material systems. We list state-of-the-art resource estimates and refer to the references therein for prior estimates. The presented gate counts are for a single run of the phase estimation circuit. The number of samples for all listed systems is 7, determined by the desired maximum failure probability, taken here as 0.1 (calculated using [792, Lemma 1] which assumes that the median value is taken from a number of incoherent repetitions of QPE. The success probability of a single run of QPE is $8/\pi^2$ [186]). QPE must be run a number of times if the overlap $\gamma \leq 1$, in general contributing a multiplicative cost of $O(1/\gamma^2)$ (though this may be reduced, as described in the main text). "Algorithm" denotes the phase estimation unitary considered (e.g., Trotterization, qubitization) as well as details about the quantization scheme (1st or 2nd), basis used, or factorization method used to compile the unitary (sparse [140], SF = single factorized [140], DF = double factorized [212], THC = tensor hypercontraction [669]). The resource estimates presented can be for different numbers of electrons and orbitals, and can have differing classical simulation complexities. As such, the results may not be directly comparable, even within a single row of the table.

Many quantum resource estimates consider active spaces with a small number of active orbitals, and so underestimate the resources required to achieve sufficiently accurate results to be informative. It is an active area of research to develop methods for increasing the resolution without increasing the basis set size, such as perturbative approaches, downfolding techniques, and embedding theories.

The end-to-end applications typically solved in the electronic structure problem can require between tens (e.g., structure determination, low temperature properties) and millions (e.g., biologically relevant molecular dynamics) of energy evaluations—each with different Hamiltonian parameters that may require preparing a new state to be measured. For example, a recent analysis of quantum algorithms applied to pharmaceutical chemistry [896] highlighted that to calculate the binding affinity between a drug molecule and its target (free energy differences) requires sampling a range of thermodynamic configurations, resulting in millions to billions of single-point energy evaluations. This introduces a large overhead when preparing a different state for each configuration and measuring its energy [805], although alternative approaches may provide more favorable scaling [941].

Comparable classical complexity and challenging instance sizes

The cost of exact diagonalization of the electronic structure Hamiltonian scales exponentially with the number of electrons and basis set size. As such, classical approaches to the electronic structure problem typically utilize a range of approximations that reduce their complexity to polynomial in the system size but introduce a (potentially uncontrolled) deviation from the exact ground state, leading to a bias in energy estimates and/or the expectation values of other observables. Approaches include Hartree–Fock (HF), density functional theory (DFT), perturbation theory, configuration interaction (CI) methods, coupled cluster (CC) methods, quantum Monte Carlo (QMC) techniques, and tensor network approaches. The cheapest approaches can be applied to thousands of orbitals but can be qualitatively inaccurate for strongly correlated systems. The most expensive approaches are more effective for strongly correlated systems, but their higher computational cost limits their applicability to roughly 100 spin orbitals. For example, [437] found that a density matrix renormalization group (DMRG) calculation performed on an 86 spin orbital active space of the cytochrome P450 enzyme molecule referenced in Table 2.2 required around 50 hours, using 32 threads, 48 gigabytes of RAM, and 235 gigabytes of disk memory. We also refer to [1046] for a comparison of 20 first-principles many-body electronic structure methods applied to a test set of seven transition metal atoms and their ions and monoxides.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Due to their extended nature, material systems are most commonly targeted with DFT. DFT can be applied to systems with thousands of electrons and orbitals but can lead to uncontrolled energy bias in strongly correlated systems. QMC and tensor network methods have been successfully applied to prototypical models of material systems and are becoming increasingly practical for more realistic models. We refer to [666, 784, 785, 900] for benchmarks of classical electronic structure methods on hydrogen chains and Hubbard models scaling to the thermodynamic limit, which act as simplified models for real materials.

Many of the techniques discussed above for computing ground and excited states of chemical systems have been extended to explicitly simulate the time dynamics of the electronic Hamiltonian. These include time-dependent HF, real-time time-dependent DFT, and time-dependent CI & CC methods. In general, the errors from the approximations made in these approaches are larger than for their static counterparts. We refer readers to [682] for a more detailed discussion of classical methods for real-time time-dependent electronic structure theory.

Speedup

It is nontrivial to determine the speedup of quantum algorithms for the electronic structure problem over their classical counterparts. If we consider the subtask of determining energy eigenstates, then for speedup greater than polynomial to be achieved, we require:

- The ability to prepare a trial state with nonexponentially vanishing overlap with the ground state as the system size increases.
- Polynomially scaling (with system size) classical algorithms having an exponential growth in their approximation parameter (e.g., bond dimension, number of excitations) as the system size increases.

Whether these two requirements can coexist in systems of interest is an active area of research [670, 144]. Even if exponential speedups are not available, it may be the case that quantum algorithms provide polynomial speedups over exact classical algorithms—and potentially over approximate classical algorithms [252].

From a complexity-theoretic viewpoint, we know that simulating the dynamics of a quantum system is a BQP-complete problem [705]. Combined with the observed difficulty of classically simulating the time evolution of electronic structure Hamiltonians, this may be taken as evidence for the possibility of an exponential speedup when simulating dynamics. In [81], quantum algorithms for simulating the fully correlated dynamics of electrons in a pseudo-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

spectral DVR or plane-wave basis [601, 77, 962] were compared against classical methods for mean-field dynamics. Large polynomial speedups were observed, ranging from superquadratic to seventh power in the salient parameters, depending on the relation between N and η.

NISQ implementations

Solving the electronic structure problem is one of the most widely studied and promoted NISQ applications. The primary NISQ approach is the variational quantum eigensolver (VQE). There have been a number of experimental demonstrations on small molecules, for example, [595, 442], as well as proposals to simulate material systems [1068, 738, 301]. Related methods, such as quantum computing–assisted QMC methods [550] have also been developed. Nevertheless, current device noise rates are too high to enable the running of circuits sufficiently deep that they can outperform classical electronic structure methods, and the number of circuit repetitions required to measure energy expectation values can be impractically large [440]. As such, there are several challenges that must be overcome if heuristic NISQ approaches are to scale to classically intractable system sizes and provide advantage over classical methods. There have also been proposals to simulate the electronic structure problem using analog quantum simulators [55], though to the best of our knowledge, these have not yet been experimentally demonstrated and are limited by the high-precision requirements of the electronic structure problem.

Outlook

Simulating the behavior of electrons in molecules and materials has repeatedly been identified as one of the most promising applications of quantum computing. Nevertheless, the discussion above highlights several challenges for current quantum approaches to become practical. Most notably, after incorporating the costs of initial state preparation and measuring observables, using larger active spaces to capture dynamic correlation, and including algorithmic repetitions to account for nonzero failure probabilities and sampling a range of parameters, a large number of logical qubits and total T/Toffoli gates may be required. The success of approximate classical methods for a wide range of chemical problems sets a high bar for quantum simulations to achieve advantage and encourages continued focus on resource estimations for end-to-end applications.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

2.2 Simulating vibrations in molecules and materials

Overview

We seek the energy eigenstates, thermal states (i.e., statistical ensembles of eigenstates at a given temperature), or dynamics corresponding to the Hamiltonian that describes the vibrations of the nuclei in a molecule or material around their equilibrium positions. This Hamiltonian contains the kinetic energy of the nuclei and the effective potential that they move on, which is determined by the electronic potential energy surface (i.e., the electronic energy expressed as a function of the nuclear coordinates). It is also possible to consider non-adiabatic couplings between the vibrational degrees of freedom and electronic ("vibronic") or rotational ("ro-vibrational") degrees of freedom.

Actual end-to-end problem(s) solved

Classically solving the Schrödinger equation while treating electrons and nuclei on an equal footing has prohibitively high computational cost for all but the smallest systems. For systems where it is valid to separate the electronic and nuclear motions (the Born–Oppenheimer approximation), we can imagine the nuclei moving on the electronic potential energy surface (PES). For molecules composed of light atoms (where relativistic effects can be neglected), the vibrations of the nuclei around their equilibrium positions provide a first-order correction to the electronic energies and influence photo-emission/absorption properties. For a system with K classical nuclei at equilibrium positions $\{R_I\}$, the vibrational Hamiltonian can be written as

$$H = -\sum_I \frac{\nabla_I^2}{2M_I} + V_e(\{R_I\}),$$

where $V_e(\{R_I\})$ denotes the nuclear potential determined by the electronic potential energy surface, obtained by first solving the electronic Hamiltonian for a range of nuclear positions. Computing vibrational eigenstates can be made classically tractable by modeling V_e as a harmonic potential, which reduces the problem to solving a number of coupled quantum harmonic oscillators. The harmonic approximation can also be used when simulating vibronic transitions between vibrational energy levels on different PESs. However, due to the coordinate change between the PESs—which acts as a squeezing and displacement operation on the vibrational modes—exact simulation is #P-hard, and therefore inefficient for both classical and quantum algorithms. Nevertheless, vibronic spectra can be efficiently approximated using classical [552, 811] and quantum [553] algorithms in many regimes of interest.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

To accurately describe nonrigid molecules or highly excited vibrational states, additional anharmonic terms are required in the potential. These can be obtained by performing higher-order fits of the potential, for example, by expanding the potential V_e to degree d. Computing accurate solutions of this Hamiltonian is prohibitively costly for many systems of interest. Probing certain phenomena, such as vibronic spectra, internal conversion, intersystem crossings, and conical intersections, additionally requires the consideration of vibrations on multiple PESs and may require a description of the nonadiabatic couplings between the different PESs (which must often be explicitly determined [828]). We seek to prepare eigenstates or thermal states, or simulate the dynamics of the anharmonic vibrational Hamiltonian, and then measure the expectation values of observables with respect to these states. Properties of interest include the following:

- The vibrational energy at the minimum of the PES, which provides a first-order correction to the electronic energies—for calculating excitation energies, determining stable molecular structures, or finding reaction pathways and rates.
- Determining transition probabilities between states and transition dipole moments—for calculating infrared/Raman spectra between vibrational levels of the same electronic state or vibronic spectra between vibrational levels of different electronic states.
- Measuring the occupancy of vibrational modes as a function of time following excitation, to understand vibrational energy transfer and relaxation in chemical systems (i.e., internal conversion and intersystem crossings).

Thermal states $\rho \propto \mathrm{e}^{-\beta H}$ (with $\beta = 1/k_B T$, where k_B is the Boltzmann constant and T the temperature) are often of greater interest in the vibrational case than in the electronic case; vibrational energies are on the order of $k_B T$ and so excited vibrational states are populated even at room temperature. This can be contrasted with the electronic structure problem, where the larger electronic energy gaps of many molecules mean that ground states are typically of primary interest at room temperature.

Dominant resource cost/complexity

A molecule with K atoms has $M = 3K - 6$ ($M = 3K - 5$ for linear molecules) vibrational modes. Each vibrational mode excitation is treated as a distinguishable particle and so the wavefunction does not need to be explicitly symmetrized. The Hamiltonian is discretized using a basis set of vibrational modal functions $\{\chi_i\}_{i=1}^{N}$, for example, the truncated eigenfunctions of the quantum

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

harmonic oscillator Hamiltonian, pseudo-spectral (Fourier) discrete variable representations, or grids.

The vibrational wavefunction can be stored in a first-quantized representation using $M \log(N)$ qubits, where the basis function of each vibrational mode is specified in binary (or an equivalent representation, such as the Gray code [897]). Alternatively, one can use second-quantized representation using MN qubits [813].

Preparing the desired eigenstate or thermal state, or simulating the dynamics can be achieved using a range of quantum algorithms, including quantum phase estimation, quantum singular value transformation, adiabatic state preparation, variational quantum algorithms, Gibbs sampling, and Hamiltonian simulation. These methods are discussed in more detail for the electronic Hamiltonian, as the explicit costs of many of these methods have not yet been determined for simulating vibrations. Nonetheless, many of the same high-level considerations apply. The complexities of subroutines to prepare eigenstates and extract observables are determined by the following observations:

(i) All methods scale as $\Omega(1/\epsilon)$ to measure the desired observable to an error of $\pm\epsilon$. For the energy, we typically seek $\epsilon \sim$ (1–10) $\text{cm}^{-1} \approx (4.56 \times 10^{-6})$–$(4.56 \times 10^{-5})$ Hartree.[4] For comparison, the largest matrix elements in the vibrational Hamiltonian (the harmonic couplings) are typically on the order of 1000 cm^{-1}, and there are $O(M)$ such terms [899]. As such, the ratio $\|H\|_1/\epsilon$ that features multiplicatively in the complexity of quantum phase estimation (at least, variants based on qubitization) can be on the order of 10^4 (or larger) for modest system sizes with $M \approx 100$.

(ii) To date, only product formula–based methods have been quantitatively studied in the context of providing coherent access to the vibrational Hamiltonian. If expanding the Hamiltonian as a linear combination of Pauli operators, the number of terms grows as $O(M^d N^{2d})$ for a degree d of anharmonic terms considered in the Hamiltonian (often at least 4th order). An alternative approach is to consider the Hamiltonian discretized by a real-space grid or Fourier pseudo-spectral basis, where the position and momentum operators can be easily applied [732, 731].

Existing resource estimates

To date, there have been no end-to-end resource estimates for the vibrational structure problem. In terms of initial steps in this direction, [897] considered

[4] Due to the close historical ties with spectroscopy, in vibrational chemistry it is common to see energies expressed as wavenumbers. Interconversion can be performed using the Planck relation.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the resources required to map vibrational operators to qubit operators, [899] compared the number and magnitude of terms in vibrational Hamiltonians to those in electronic structure Hamiltonians, and [995] estimated the number of terms and Trotter steps required to perform quantum phase estimation on polyyne molecules.

Caveats

Quantum algorithms and many (but not all [1092]) classical algorithms for simulating the vibrations of nuclei require the availability of an electronic PES, from classical calculations. For a grid-based interpolation of the multi-dimensional PES with h points per dimension, we require $O(h^M)$ PES evaluations. Nevertheless, several interpolation techniques and adaptive methods have been developed to obtain high-accuracy PESs, at lower costs. Moreover, a few molecules with classically challenging vibrational spectra have been identified with classically easy-to-compute electronic PESs [899].

There has been less work on the number of vibrational basis states required to achieve a given accuracy than in the electronic case. While rigorous results exist for more simple bosonic Hamiltonians [991], the truncation level N has not yet been established for anharmonic potentials.

Comparable classical complexity and challenging instance sizes

A hierarchy of approximate classical methods has been developed for computing vibrational eigenstates, which trade increased accuracy for increased cost. Vibrational states with a multireference nature—which are required to describe vibrational resonances that arise due to near degeneracies between different vibrational eigenstates, resulting from anharmonicities in the PES—require more accurate (and thus costly) methods. Moreover, nonrigid molecules require a higher-degree approximation of the PES, leading to an increased cost for classical methods—and potentially increasing the complexity of the resulting eigenstates. For such challenging systems, accurate classical results have been obtained for molecules with $G = 20$–30 atoms [237, 84, 989, 99].

Recently, a number of quantum-inspired classical algorithms (see [811, 810] and references therein) have been developed for classically approximating the results of Gaussian boson sampling experiments. These experiments have been proposed as analog simulators of harmonic vibrational phenomena, including vibronic spectra and vibrational dynamics (see §NISQ implementations, below). This reduces the regime where quantum advantage may be possible and reinforces the necessity of considering anharmonicities in the vibrational Hamiltonian [811].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

For a review of classical algorithms for simulating coupled vibrational and electronic degrees of freedom, we refer to [1092, 597]. Commonly used algorithms include multiconfigurational time-dependent Hartree (MCTDH) and the ab initio multiple spawning method (AIMS).

Speedup

The speedup for quantum algorithms in computing vibrational eigenstates depends on the overlap and error convergence of classical trial states to the true eigenstate and energy. This has yet to be determined for systems of interest. Nevertheless, in spectroscopic calculations that start from a classically easy-to-prepare state, the overlap between the initial state and a desired excited state becomes a quantity of interest—corresponding to the absorption intensity—rather than a limiting factor on the calculation. For example, in [898] it was proposed to use quantum phase estimation to project from the initial state into other eigenstates with probability given by the squared overlap between the states. However, while a single (exponentially costly) classical diagonalization of the vibrational Hamiltonian would provide complete access to the entire vibrational spectrum, a large number of repetitions of the quantum algorithm would be required to reconstruct the spectrum.

As discussed above, the development of recent quantum-inspired algorithms [811, 810] has reduced the prospect of achieving quantum advantage for vibrations in harmonic potentials, motivating the need to include anharmonicities in the vibrational Hamiltonian, or to consider more complex models that nonadiabatically couple vibrational and electronic degrees of freedom. We refer to [597] for a discussion of the classical complexity of simulating such models.

NISQ implementations

There have been proposals to apply variational algorithms to solve the vibrational structure problem [750, 813, 897, 899], but additional developments are required in order to implement sufficiently deep circuits to surpass classical methods, and the number of circuit repetitions required to measure energy observables is likely a bottleneck [899].

There have also been several proposals and experimental demonstrations for using analog quantum simulators to simulate molecular vibrations. Analog simulations have been performed for zero and finite temperature vibronic spectra [553, 1015] as well as vibrational dynamics [946]. We note that these approaches use harmonic approximations for the vibrational potential, and can be approximated efficiently by classical algorithms in some regimes [811]. There have also been analog quantum simulations of systems with coupled electronic

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

and vibrational degrees of freedom (typically via a linear vibronic coupling model) including simulations of conical intersections [1016, 1036, 1001] and vibrational assisted energy transfer [443].

Outlook

Further work is required to identify target systems that are challenging to simulate classically, but that may be amenable to quantum algorithms. In addition, existing quantum algorithms need to be further optimized for the accuracy required in vibrational structure problems and the form of the vibrational Hamiltonian. This will enable resource estimates for end-to-end applications, such as estimating vibrational spectra or simulating vibrational dynamics.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

3

Nuclear and particle physics

Simulating nuclear and particle physics is an inherently quantum problem. There have been proposals to use quantum computers to accelerate simulations of quantum field theories, nuclear physics, neutrino physics, and quantum gravity [104]. In this chapter, we focus on the simulation of quantum field theories and nuclear physics, as these have received the most attention in the literature to date and are the closest to having end-to-end resource estimates available. While not covered explicitly in this chapter, the building blocks of quantum algorithms for data analysis in high-energy physics [340] can be found in Chapter 20 on variational quantum algorithms and Chapter 9 on machine learning. For existing reviews of quantum computing for nuclear and particle physics, we direct the reader to [104, 844, 92, 404, 632, 105, 110].

The authors are grateful to Zohreh Davoudi and John Preskill for reviewing this chapter.

3.1 Quantum field theories

Overview

We seek the static and dynamic properties of quantum field theories, specifically gauge field theories and scalar field theories. Gauge field theories describe the interactions between matter and/or gauge degrees of freedom and can be classified by their symmetry groups, such as U(1) (describing quantum electrodynamics), SU(2) (the weak interaction), and SU(3) (quantum chromodynamics). Scalar field theories describe interactions between scalar fields, such as the Higgs field or ϕ^4 theory.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Interacting quantum field theories are typically not analytically solvable, and techniques such as perturbation theory are only accurate in some parameter regimes. For example, low-energy quantum chromodynamics (QCD), relevant to quark confinement and hadron formation, cannot be treated perturbatively. As such, complex scattering processes at particle accelerators are currently treated with a combination of first-principles calculations and approximate phenomenological methods.

To tackle quantum field theories numerically from first principles, lattice field theory is employed. The Lagrangians arising from lattice field theory can be numerically solved on classical computers using Euclidean Monte Carlo methods, which have proven highly efficient and accurate for a number of static quantities, including hadron masses, static matrix elements of (primarily) time-local operators between hadronic states, and even certain properties of light nuclei [332, 333, 44]. However, these classical Monte Carlo methods become intractable due to a sign problem in two regimes: (i) at high fermion density (of considerable scientific interest for understanding the decomposition of neutron stars and large atomic nuclei) and (ii) in simulations of real-time dynamics (e.g., scattering problems). Hamiltonian formulations of these problems are challenging due to the size of the required Hilbert space. As such, there have been a number of proposals to use quantum computers for calculating the static and dynamic properties of matter described by scalar and gauge field theories. For further background, see [844, 104, 761] and references therein.

Actual end-to-end problem(s) solved

Classical computational methods for lattice field theories have produced a number of insights, including high-precision computations of fundamental quantities (such as the muon's magnetic moment and quark masses), tests of beyond-the-Standard-Model physics (such as charge conjugation and parity (CP) violation and beyond-Higgs theories), and nuclear cross sections with dark matter candidates or neutrinos. For a more complete and detailed list, we refer the reader to [333, Page 6].

We primarily focus on the case of lattice field theories in the Hamiltonian formulation, which explicitly separates temporal and spatial degrees of freedom [640] and discretizes the d-dimensional space using an L^d lattice (which may be noncubic). Matter degrees of freedom (e.g., quarks, scalar fields) are placed on the vertices of the lattice. Gauge degrees of freedom (e.g., the value of the electromagnetic field) are placed on the links between lattice sites. Dynamical simulations proceed by initializing the system in a desired state [82], performing time evolution under the Hamiltonian, and measuring relevant observables. See [588] for an example of simulating scalar field theories. Static

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

simulations aim to prepare a state of interest, such as the ground state of a collection of quarks representing a composite hadron, and then measure observables of interest, including binding energies, as well as structural and reaction properties.

Quantum simulations of lattice field theories may be incorporated as part of a larger (multiscale) computational workflow. For example, when studying scattering processes (such as those that occur at particle accelerators), it is not necessary to simulate the entire scattering process on a quantum (or classical) computer. Instead, the scattering cross section can be separated into short- and long-distance contributions, which can be computed using perturbative and nonperturbative (e.g., quantum or classical simulation of lattice gauge theories) methods, respectively [413, 103].

Dominant resource cost/complexity

In this section, we focus predominantly on the simulation of dynamics in lattice gauge theories (LGTs), as the majority of studies to date have considered this application. In the standard formulation, one allocates one qubit per fermion (or antifermion) type per site of an $N = L^d$ lattice. Each gauge degree of freedom (one in U(1), three in SU(2), eight in SU(3)) requires its own register associated with each edge between lattice sites. The quantum numbers associated to the gauge degrees of freedom are encoded in binary, up to a maximum cutoff value Λ, so the corresponding register requires $\log(\Lambda)$ qubits. It was shown in [991] that for time evolution performed with fixed lattice spacing, the cutoff can be set as $\Lambda = \Lambda_0 + \widetilde{O}(T) \cdot \text{polylog}(N/\epsilon)$, where Λ_0 is the maximum initial value of the gauge fields, T is the time evolution duration, and ϵ is the resulting error in the final state. Hence, the overall number of qubits required to store the state of the system scales as

$$O\left(L^d \log\left(\Lambda_0 + T \ \text{polylog}\left(\frac{L^d}{\epsilon}\right)\right)\right).$$

Algorithms for implementing time evolution under LGT Hamiltonians are presented in [991, 927, 593, 852, 292, 334, 873]. It is necessary to (approximately) maintain gauge invariance during the simulation, which can be achieved either by the choice of formulation, by actively protecting symmetries [487, 994], or by detecting and eliminating the gauge-violating states [960, 864]. As an example of the first option, one can calculate the desired Hamiltonian matrix elements on the fly using Clebsch–Gordon coefficients [213], but this is expensive in terms of elementary quantum operations [593]. An alternative approach described in [292] encodes only the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

physical transitions in the SU(3) gauge theory. This method requires many controlled operations and a large classical precomputing overhead.

Existing resource estimates

The number of T gates required to simulate instances of the lattice Schwinger model (U(1) LGT in $d = 1$ with both matter and gauge degrees of freedom) was studied in [927]. That work considered the resources required to perform Trotterized time evolution and estimate the electron-positron pair density. The most complex simulations analyzed (64 lattice sites, cutoff of $\Lambda = 8$) required 5×10^{13} T gates per shot, and 333 logical qubits. Such a circuit would need to be repeated $O(1/\epsilon^2)$ times to estimate the pair density to accuracy ϵ; this overhead could be improved to $O(1/\epsilon)$ using quantum amplitude estimation at the expense of increased gate depth [927]. These estimates were later improved in the small system-size, long-time, or low-error regime using algorithms based on qubitization with quantum signal processing [892]. Note that a simulation of the 64-site lattice Schwinger model with $\Lambda = 8$ is well within the range of classical simulations [389, 734].

Reference [593] performed similar resource estimates for the simulation of dynamics in U(1), SU(2), and SU(3) LGTs for $d = 3$. These resource estimates were performed for synthesizing the time evolution operator, with choice of simulation parameters inspired by the following tasks: computing transport coefficients relevant to the study of quark-gluon plasmas, simulating heavy ion collisions, and computing the hadronic tensor of the proton—although we note that the costs of initial state preparation and observable measurements are not included in these resource estimates. Logical qubit counts ranged from 10^4 to 10^8, while T gate counts ranged from 10^{17} to 10^{56}. The large constant factors present in these resource estimates stem partly from the use of quantum arithmetic (e.g., constituting 99.998% of the gate count in the most expensive calculations [593]), and partly from the decomposition of the plaquette term, which is exponential in the number of colors. The large gate counts also arise from the value chosen for the error in the time evolution operators $\epsilon = 10^{-8}$, which may be overly conservative when viewed in conjunction with other sources of error.

These gate counts can be reduced significantly using the approach of [334], which investigated alternative ways of representing the simulation, allowing for an improved grouping of terms in the product formula. These ideas were illustrated for SU(2) in 1+1D, but can likely be generalized to more complex, higher-dimensional LGTs.

Nevertheless, we note that any implementation scaling as $\Omega(TL^3)$ (i.e., linearly in spacetime volume) already faces a factor of 10^8 for $T = L = 100$,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

highlighting the potentially large resource counts of simulating quantum field theories. Recent work presented an algorithm for simulating the time evolution of LGTs that achieves this optimal complexity, up to polylogarithmic factors [873]. This work uses a number of subroutines for Hamiltonian simulation [479, 718, 615] and requires locality-preserving fermion- and boson-to-qubit mappings. Resource estimates were carried out for U(1), SU(2), and SU(3) LGTs. For simulating time evolution in an SU(3) LGT on a lattice with $T = L = 100$, the algorithm required approximately 10^{21} T gates and 6×10^7 logical qubits. Despite the large improvements compared to [593], the significant discrepancy from the expected lower bound of 10^8 discussed above suggests that there is further opportunity for optimization in the implementation of algorithms for simulating LGTs.

Caveats

Additional investigation is required to better quantify the theoretical uncertainties arising from discretization, finite-volume, and Hilbert space truncation effects of quantum computing formulations, as well as the algorithmic errors present in quantum simulation algorithms applied to lattice and scalar field theories.

For example, discretization of the continuous field theory to the lattice setting introduces a number of nuances (which are also present in classical approaches but must be considered afresh in quantum calculations). As discussed in [970, 598], discretization of the fermion field breaks the Lorentz invariance of the fermion kinetic term, which introduces unphysical additional flavors of fermions (known as the fermion doubling problem). This issue can be mitigated in several established ways, each with their own merits and drawbacks for quantum simulation [747]. It is also necessary to carefully track other errors resulting from discretization and ensure that these vanish when scaling and extrapolating to the continuum limit [588, 231].

As noted in [104, Section 6b] and [292, 105], there are a number of possible representations/basis sets that can be used for the gauge degrees of freedom, and it is currently unclear which choice is optimal for quantum simulation.

Comparable classical complexity and challenging instance sizes

The end-to-end scattering processes typically considered at particle accelerators are too complex to be solved from first principles and are instead tackled using a range of approximate techniques [413]. These computations often include parameters obtained from first-principles LGT calculations on simpler systems, and they typically proceed through a Lagrangian formulation, rather than a Hamiltonian formulation. This leads to Monte Carlo sampling of a path

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

integral in Euclidean spacetime, the application of which to dynamical problems or static problems with high fermion density is limited by the fermionic sign problem [793]. For example, the phase diagram of QCD and the existence of exotic phases at extreme densities, in and out of equilibrium, have eluded classical methods. Nevertheless, classical approaches have been very effective for static problems with low fermion density and for dynamical scattering problems at low energy and low inelasticity; for a review of current state-of-the-art computations and limitations, see [333, 585] and companion whitepapers referenced therein.

Recent work has begun to investigate using tensor network methods to simulate the Hamiltonian formulation of LGTs; see, for example, [389] ($d = 2, L = 16$, U(1) LGT with gauge field cutoff $\Lambda = 1$) and [734] ($d = 3, L = 8$, U(1) LGT with gauge field cutoff $\Lambda = 1$). Like quantum simulations, tensor network approaches are sign-problem free and thus may be of interest in regimes out of reach of conventional Monte Carlo–based approaches. However, tensor network approaches are currently limited to small system sizes, and often need to be verified by comparing to other methods, as they do not come with provable guarantees on the bond dimension required for capturing the entanglement structure of the states present in LGTs. For recent reviews on the use of tensor networks to simulate LGTs, we refer the reader to [91, 735].

Speedup

For simulations with a sign problem, classical Monte Carlo methods are exponentially costly in system size [997]. In addition, it was observed that the bond dimensions required for tensor network approaches increase rapidly with system size [734], suggesting the potential for exponential quantum speedups for dynamical problems. This suggestion is reinforced by the BQP-completeness of the simulation of certain field-theoretic processes [589]. Nevertheless, the constant prefactors for quantum simulations of LGTs are currently high, and we require the (currently underexplored) ability to efficiently prepare initial states of interest.

NISQ implementation

A number of works have investigated the simulation of scalar or gauge field theories on noisy digital, or analog, quantum simulators. A common strategy is to map the lattice field theory Hamiltonian to that of a bosonic system, such as cold atoms or trapped ions; see, for example, [104, 105, 632, 92] and references therein. While these techniques appear promising for simple Hamiltonians, such as the Schwinger model, it may be challenging to engineer the more complicated interactions required in nonabelian gauge field theories. There

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

have also been works applying variational algorithms to lattice field theories, such as [641, 67, 699], as well as digital simulations of the time dynamics of the lattice Schwinger model [631, 387].

Outlook

Investigations into how quantum computers can be used to complement classical methods for simulating lattice field theories are advancing rapidly. While quantum computers can, in principle, efficiently simulate the complex scattering experiments performed in particle accelerators, the resources required to do so would be impractical using currently known techniques. Future work must determine the best targets for quantum simulations and reduce asymptotic scaling as well as constant prefactors. In particular, the qubit encoding (currently scaling as $O(L^d)$ qubits for a lattice in d spatial dimensions with each dimension having L sites) means that a large number of logical qubits will likely be required for computations of interest where, as illustrated by examples above, we may consider $L = 10$–100 to challenge classical approaches.

3.2 Nuclear physics

Overview

Nuclear physics describes the behavior of individual nuclei, as well as that of dense nucleonic matter, such as neutron stars. The structure of nuclei can be approximately described using the shell model (see [337] for an overview), a phenomenological model with parameters fitted to experimental observations. However, high-accuracy descriptions of nuclear structure, exotic nuclei, accurate scattering cross sections, and nonequilibrium phenomena require a first-principles treatment. Describing the properties of nuclei from first principles (e.g., lattice quantum chromodynamics simulations) is beyond the reach of analytic and current computational capabilities for all but the simplest nuclei [369, 108, 332]. Nevertheless, one can often integrate out the short-range physics to obtain effective field theories (EFTs) that describe the interactions of nucleons. The prototypical example is chiral effective field theory, which describes the interactions of nucleons and pions (pions, which have mass less than 6× smaller than that of the proton, are mediators of the residual strong interaction between pairs of nucleons). The parameters of the EFT can be inferred from experiments or directly from lattice quantum chromodynamics (QCD) calculations, resulting in a many-body Hamiltonian that describes the formation, structure, and potential decay of nuclei.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Actual end-to-end problem(s) solved

An EFT provides a many-body Hamiltonian describing how nucleons interact. This quantum many-body problem can be tackled using a range of classical methods, which employ different mathematical approaches to approximately solve the nuclear many-body Schrödinger equation including quantum Monte Carlo (QMC) methods [232], the no-core shell model (NCSM) [100], the coupled cluster (CC) method [485], the self-consistent Green's function (SCGF) method [230], the in-medium similarity renormalization group (IM-SRG) method [958], and nuclear lattice methods [658] (a related review article discussing inputs to these calculations is given by [987]).

A common problem is to prepare the ground state of a collection of nucleons, in order to compute nuclear binding energies and determine if a given nucleus is stable (e.g., determining the long lifetime of ^{14}C [740, 485]). Simulations can also be used for computing scattering cross sections, in order to analyze experiments on nucleus-neutrino scattering [877], beta decay [472], and nuclear reactions. Reactions, such as nuclear fission and nuclear fusion, can also be studied using explicitly time-dependent approaches [115], although these have higher computational costs than static computations and are often based on semiclassical, mean-field, or other phenomenological models. Simulating both fusion and fission reactions has a number of use cases, such as an improved understanding of nuclear astrophysics, where reactions commonly occur at energies too high or too low to be replicated in experiments [796].

Dominant resource cost/complexity

The prototypical EFT for nuclear interactions is chiral effective field theory. At very low energies, the theory can be expressed as a convergent perturbative expansion (chiral perturbation theory). However, for the larger energies relevant to multi-nucleon systems, the theory becomes nonperturbative. Despite several notable successes, developing EFTs that converge across a wider range of scenarios remains an active area of research [987, 489].

At low energies (below the rest mass of the pion), it can be appropriate to apply pionless EFTs, which integrate out the pions, leading to implicit interactions between nucleons, including a 3-nucleon contact interaction required at leading-order by renormalization. At higher energies, it is necessary to explicitly account for the effect of pions. Pionfull EFTs are typically studied numerically, as it can be difficult to obtain analytic predictions [987]. In a formulation known as the one-pion-exchange Hamiltonian, pions are integrated out, leading to a long-range two-body interaction between nucleons, which decays exponentially with distance. In an alternative formulation, known as dynam-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ical pion EFT, the (relativistic) pions and their interaction with nucleons are explicitly simulated.

A common formulation of EFT simulations is to project the problem onto a lattice in position or momentum space [667]. For quantum simulations formulated on a lattice, a typical second-quantized mapping uses 4 qubits per lattice site for nucleons (two isospin degrees of freedom and two spin degrees of freedom), although additional qubits may be required if using a fermion-to-qubit mapping that maintains the locality of the fermionic Hamiltonian (e.g., 6 qubits per site [1028]). If simulated explicitly (e.g., dynamical pion EFT), the value of the spin-0 pion field at each lattice site can be stored using a number of qubits scaling logarithmically with the pion energy cutoff by storing its quantum number in binary.

An alternative approach is to project the EFT Hamiltonian onto a single-particle basis, commonly harmonic oscillator eigenfunctions [153]. In second quantization, a qubit is required per single-particle (iso)spin mode. However, this mapping can lead to long-range interactions between modes, and in the most general case, up to $O(N^6)$ distinct terms for an N-mode system [104, 957].

Quantum algorithms that prepare energy eigenstates (or good approximations thereof) scale either as $1/\gamma$ (where γ is the overlap of the initial state with the desired eigenstate) [688], or with the minimum gap size along an adiabatic or thermalizing path. If we are only interested in measuring the energy of the state, this can be obtained using the quantum phase estimation algorithm, which also projects the system into the corresponding energy eigenstate. The cost of this approach scales as $O(1/\gamma^2)$ in terms of the original overlap γ, which can be improved to $O(1/\gamma)$ using amplitude amplification at the expense of increased circuit depths. Once the desired state has been prepared, observables can be measured to precision ϵ with complexity $O(1/\epsilon^2)$ (direct sampling) or $O(1/\epsilon)$ (using amplitude estimation, also requiring coherent state preparation, e.g., via amplitude amplification).

The above algorithms for preparing states (and related algorithms for performing time evolution in dynamics simulations) require access to the Hamiltonian, typically implemented via block-encoding or Hamiltonian simulation.

Existing resource estimates

The gate costs of Hamiltonian simulation using product formulas for neutrino-nucleus scattering in pionless EFT were studied in [877]. The T-gate costs of Hamiltonian simulation using product formulas, as well as quantum phase estimation, were estimated in [1028] for pionless EFT, one-pion-exchange EFT, and dynamical pion EFT (all formulated on a lattice). As an example, resources to simulate the dynamics of 40 nucleon system ranged from 4×10^{12} T gates

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

and 6000 logical qubits for pionless EFT, to 2.0×10^{24} T gates and 6000 logical qubits for one-pion-exchange EFT, to 5.2×10^{49} T gates and 168,000 logical qubits for dynamical pion EFT (all to an error $\epsilon = 0.1$). The resources required for the latter EFT are considerably higher, due to the qubit overhead and worse error dependence that stems from explicitly simulating the pions. Resource estimates for other parameter regimes can also be found in [1028], where it is noted that higher-order EFTs are required for the accurate simulation of medium- and large-mass nuclei, which may further increase simulation costs.

Caveats

Developing accurate nuclear EFT Hamiltonians describing heavier nuclei is an active research direction. Most studies are currently based on phenomenological models, which are limited in their predictive capabilities. In some systems, some higher-order EFTs provide sufficient accuracy, but only after fitting the EFT coefficients to experimental data. We refer the interested reader to [987] for a more detailed discussion.

Comparable classical complexity and challenging instance sizes

Classical approaches use similar techniques to those developed for the electronic structure problem, such as perturbation theory, Monte Carlo methods, or coupled cluster approaches. References [232, 100, 485, 230, 958, 658, 523] provide excellent overviews of state-of-the-art approaches. Classical methods can provide outstanding agreement with experiments for the binding energies of small nuclei with 20–50 nucleons [523]. As a further example, recent high-accuracy simulations of the ^{100}Sn nucleus have improved the agreement between theory and experiment for observed β-decay rates [472]. Time-dependent simulations of dynamics or nonequilibrium phenomena are more challenging and are an active area of research [115, 796]. We refer readers to [233] for a detailed analysis of the capabilities and requirements of exascale supercomputers in nuclear physics simulations.

Speedup

The majority of classical approaches for the nuclear structure problem are designed to run in polynomial time with respect to the system size but introduce errors due to the use of approximations (e.g., truncating the expansion in coupled cluster methods) [523]. For quantum computers to achieve exponential speedups, one needs to identify systems where (i) classical methods require an exponential increase in resources to obtain accurate results and (ii) it is efficient to prepare an initial state for the quantum computation with at least in-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

verse polynomially large (in terms of the system size) overlap with the desired state. Recent initial investigations have explored whether these requirements coexist in chemical systems [670]. We are not aware of similar work in nuclear physics, although it has been noted that the states obtained from classical methods could be used as initial states for quantum algorithms [110].

For simulating the dynamics of nuclear systems, classical methods typically proceed via mean-field methods, and exact simulations are limited to small system sizes due their exponential scaling. In contrast, quantum algorithms are able to simulate the dynamics of nuclear systems with a cost scaling polynomially with the system size and desired accuracy.

NISQ implementation

Almost all of the work to date on demonstrating near-term quantum computing approaches for the nuclear structure problem has focused on variational algorithms, such as [366, 724, 955, 929], or small-scale simulations of dynamics [877, 1000]. There is currently no evidence that near-term quantum devices will be able to implement sufficiently deep circuits to achieve advantage over their classical counterparts with these methods.

Outlook

Nuclear physics presents a classically challenging quantum many-body problem that appears well suited to simulation on quantum computers. While there are similarities to the electronic structure problem in quantum chemistry, which has led to a transfer of several ideas, nuclear Hamiltonians are typically more complex, involving (depending on the formulation) long-range interactions, 3-body interactions, multiple species of nucleons and pions, and interactions with nontrivial spin and isospin dependence. The simulation of nuclear reaction dynamics appears a particularly interesting target, and future work should determine the resources required for end-to-end simulations, including state preparation and measurement of observables.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

4

Combinatorial optimization

Combinatorial optimization problems are tasks where one seeks an optimal solution among a finite set of possible candidates. In industrial settings, combinatorial optimization arises via scheduling, routing, resource allocation, supply chain management, and other logistics problems, where it can be difficult to find optimal solutions that obey various desired constraints. The field of operations research—which came to prominence after its application to logistics problems faced by World War II–era militaries—applies methods of combinatorial optimization (as well as continuous optimization) to these problem areas for improved decision-making and efficiency in real-world problems.

Combinatorial optimization problems are also at the heart of classical theoretical computer science, where they are used to characterize and delineate complexity classes, such as P and NP. Typical combinatorial optimization problems have limited structure to exploit, and therefore quantum computing most often only provides polynomial speedups. In fact, it came as a surprise in the early days of quantum computing research that for a wide variety of such problems, quantum computers do offer up to quadratic speedups via Grover's search algorithm [464]. Subsequently, much effort was devoted to understanding how Grover search and its generalization, amplitude amplification, offer speedups for various combinatorial optimization problems.

In this chapter, we cover several distinct approaches to solving combinatorial optimization problems. First, we look at combinatorial optimization through its relation to search problems, where Grover's algorithm, or its generalizations, can be applied to give a quadratic or subquadratic speedup. Then, we cover several proposals—variational algorithms (viewed as an exact algorithm), the adiabatic algorithm, and the "short-path" algorithm [505, 329]—that have the potential to surpass the quadratic speedup of Grover's algorithm. We discuss the (limited) evidence that these approaches could generate significant advantages, as well as the associated caveats.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

We do not specifically cover the large body of work on quantum approaches for *approximate* combinatorial optimization (typically variational quantum algorithms or quantum annealing). These algorithms usually translate the optimization problem to energy minimization of a spin system with a Hamiltonian that encodes the classical objective function. They apply some physically motivated heuristics to efficiently generate solutions that have low energy, and seek a better objective value than could be generated classically in a comparable amount of time. An advantage of these approaches is that they are often more compatible with noisy near-term hardware. While approximate optimization remains an interesting direction, these quantum algorithms are heuristic and there is a general scarcity of concrete evidence that they will deliver practical advantages.

We refer the reader to [6] for a comprehensive survey of quantum methods for combinatorial and continuous optimization.

The authors are grateful to Ashley Montanaro for reviewing this chapter.

4.1 Search algorithms à la Grover

Overview

Grover's search algorithm [464] and its generalizations, such as amplitude amplification, are essential sources of quantum speedups. A straightforward application of Grover search in the spirit of optimization is quantum minimum finding [367, 48], which provides a quadratic speedup for finding the minimizer of a function on a given set of elements.

As search is a generic primitive, Grover's algorithm is widely applicable, and it can speed up many subroutines, especially in algorithms for combinatorial optimization. We list a few representative applications that demonstrate how Grover's algorithm may be applied to speed up combinatorial optimization.

Actual end-to-end problem(s) solved

The goal is to solve a search problem, that is, decide whether there is an element among a set of objects that satisfies some criterion, and if there is such an object, find one. Many combinatorial optimization problems are fundamentally search problems; a notable class of examples are graph problems, such as finding a maximal independent set, a k-coloring, a lowest weight Hamiltonian

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

cycle[1] (called the traveling salesperson problem), or the shortest path between two vertices.

For conceptual clarity, here, we focus on the prototypical Boolean satisfiability problem, that is, SAT solving: given a Boolean formula in the so-called *conjunctive normal form*, decide whether it has a satisfying Boolean assignment (and if so, find one). A formula in this form consists of some constraints (called *clauses*) each containing the logical AND of some variables or their negation (called *literals*). We denote the number of Boolean variables by n and the total number of literals of the formula by ℓ (typically $\ell \geq n$ since each variable should appear at least once).

Dominant resource cost/complexity

If there are at least m marked elements among N possible ones, then the search problem can be solved with high probability by using $O(\sqrt{N/m})$ Grover iterations. Each Grover iteration requires generating a uniform superposition over the N elements starting from the all $|0\rangle$ state and checking whether an element is marked (in superposition), which can be implemented with gate cost $O(\ell+n)$. If the formula is satisfiable, then there is at least one solution, thus $O(\sqrt{2^n})$ Grover iterations suffice, giving an overall complexity of $O((\ell + n)\sqrt{2^n})$.

In some applications, it is useful to consider a generalization of Grover search, amplitude amplification, which enables working with an arbitrary prior distribution on the elements, unlike Grover's algorithm which effectively uses a uniform prior. The relevance of this extension can be seen through the example of 3-SAT, which is a restricted version of SAT where each clause has at most 3 literals. A clever application of amplitude amplification described by Ambainis [24] for solving 3-SAT more efficiently uses Schöning's algorithm [908] and thus generates a nontrivial prior distribution on the solutions.

The complexity of amplitude amplification is similar to that of Grover search in general. If $|\psi\rangle$ is the quantum state representing the prior distribution, so that measuring the state yields a marked element with probability at least p, then $O(\sqrt{1/p})$ "Grover iterations" suffice to find a marked element with high probability. The algorithm requires preparing the initial state $|\psi\rangle$, and then each iteration consists of a reflection $2|\psi\rangle\langle\psi| - I$ around $|\psi\rangle$ and checking whether an element is marked (in superposition). The former reflection can be implemented with two uses of the circuit that prepares $|\psi\rangle$ from the all $|0\rangle$ state, and a reflection about the all $|0\rangle$ state.

[1] A Hamiltonian cycle in a graph is a cycle that visits each vertex once, not to be confused with a quantum Hamiltonian.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Existing resource estimates

There are several studies on the resource estimation of Grover-type (sub)quadratic speedups. Due to the wide range of these problems, we do not focus on explicit gate counts on any particular problem/implementation variant, but rather list some prominent articles and illustrate their findings at a high level [223, 895, 79, 217, 216, 532]. Unfortunately, these recent studies revealed that quadratic or smaller speedups alone are unlikely to be useful for the foreseeable future, unless the large overheads of current fault-tolerant quantum computing schemes can be greatly reduced. For example, [223] concluded that even if there is some reasonable advantage in quantum gate counts for solving the constraint satisfaction problems that they consider, the classical computation supporting the fault-tolerant quantum computation actually voids the speedup in practice. They state that "Even when considering only problem instances that can be solved within one day, we find that there are potentially large quantum speedups available. ... However, the number of physical qubits used is extremely large, ... In particular, the quantum advantage disappears if one includes the cost of the classical processing power required to perform decoding of the surface code using current techniques." The most recent of the references listed above [532] estimates that achieving a quantum advantage via a quadratic speedup requires at least a month-long computation already if each iteration contains at least one floating-point operation. The situation looks more promising for cubic and quartic speedups, but unfortunately such improvements seem to require techniques beyond Grover search.

Caveats

Grover originally described his result as "A fast quantum mechanical algorithm for database search" [464]. If we work in the circuit model of quantum computation, then strictly speaking Grover search gives a slowdown for database search, as every Grover iteration needs to "touch" every element in the database. If we anyway need to touch all N elements in the database, then the best we can do is to simply go over every element in linear time $O(N)$. Grover's search circuit, on the other hand, would have gate complexity $\widetilde{O}(N^{3/2})$, clearly worse than sequentially going through the entire dataset.

In the database scenario, we can only recover the quadratic speedup if we assume that we can use a quantum random access memory (QRAM), with constant (or logarithmic) cost for each database query. The analogous assumption regarding ordinary RAM is often made in classical computer science, simply because RAM calls are cheap in practice. However, since a RAM call should be able to touch every bit of the database, from a circuit complexity perspective

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

a RAM call must have gate cost at least N. On the other hand, from a time complexity perspective, one can view a RAM call as a massively parallel piece of computation implementable in a binary tree structure with logarithmic depth.[2] While QRAM can also be implemented with a quantum circuit of $O(\log(N))$ depth, a similar accounting might not be fair in the quantum case, depending on the eventual cost of hardware implementation—especially if error correction of the QRAM is necessary and the entire QRAM circuit is implemented in a fault-tolerant fashion.

Nevertheless, Grover's algorithm can provide a quadratic speedup without extra hardware assumptions when the elements of the list that we search over can be easily generated and checked "on the fly." For example, in the case of SAT, we search over the 2^n possible truth assignments, yet we can easily check whether an individual assignment is satisfactory by simply substituting the assignment into the formula and evaluating the resulting Boolean expression. This is the defining feature of problems in the complexity class NP, whose solutions are efficient to verify.

Comparable classical complexity

For the unstructured search problem, exhaustive search is essentially the best that can be done, with a running time $\sim \ell \cdot 2^n$. Of course, SAT seems to be far from unstructured, but under the Strong Exponential-Time Hypothesis [558, 220] the best classical algorithm for SAT has running time $2^{n-o(n)}$.

A similar argument holds for the generalized problem considered in the setting of amplitude amplification: if we have some prior distribution, we can classically find a marked element by sampling from this distribution roughly $1/p$ times. Since unstructured search is a special case of this problem, we cannot hope for a better classical algorithm in general.

Speedup

The speedup is quadratic in terms of the number of required iterations if we compare to corresponding naive classical algorithms. It can be shown that this speedup is optimal in the black-box query model [122]. Moreover, we do not expect that there would be a bigger than quadratic speedup in gate complexity [210] in the general (non-black-box) case.

[2] Viewing RAM as a low-depth circuit disregards issues regarding signal transmission. Considering that the speed of light is finite and we have only 3 dimensions to fit the memory cells into, a RAM call should asymptotically cost at least $\sqrt[3]{N}$ time. In fact, state-of-the-art clock speeds are already in a regime where the speed of light may be a bottleneck, so we might eventually need to reconsider how the time complexity of RAM is modeled.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Outlook

We have discussed how Grover search provides a quadratic speedup for SAT, and how amplitude amplification yields a quadratic speedup for Schöning's 3-SAT algorithm [908]. Since the best known 3-SAT solvers [526, 497] have complexity $O(1.308^n)$—only slightly better than Schöning's $O(1.334^n)$ complexity—this implies a close-to-quadratic quantum speedup. However, note that this relates to worst-case complexity, and on practical instances, the scaling can be much better.

We now comment on some of the other combinatorial optimization problems where Grover's algorithm can be used as a subroutine. One class of examples is graph-related problems. In the literature, these problems are most often studied in the query model, therefore, here we also only discuss their speedup in terms of query complexity. (Since these are (sub)quadratic speedups, we know that the fault-tolerant resource estimates will be unfavorable anyway, as discussed above.) For instance, the problem of finding the shortest paths from a single source s in graph $G = (V, E)$ to all other vertices $v \in V$ can be solved classically using Dijkstra's algorithm in time $O(|E| + |V| \log |V|)$ if the graph is provided with its adjacency list (and with query complexity $O(|E|)$), whereas the quantum query complexity of this problem is $\widetilde{\Theta}(\sqrt{|V||E|})$ [368]. Reference [368] determines the query complexity of several other graph problems such as deciding graph connectivity and strong connectivity as well as finding the minimum-weight spanning tree. For all of these problems, there is a similar (sub)quadratic quantum speedup.

One graph problem that is often mentioned in connection to quantum computation is the (in)famous traveling salesperson problem. However, for this problem, the best provable speedup is only subquadratic. The naive classical algorithm runs in time $\widetilde{O}(n!)$, and Grover's algorithm offers a quadratic speedup over it. The best classical algorithm uses dynamic programming and runs in time $\widetilde{O}(2^n)$. Ambainis et al. [28] showed how to obtain a speedup over this algorithm by combining classical precalculation with recursive applications of Grover's search resulting in time complexity $\widetilde{O}(1.728^n)$ assuming that QRAM calls have unit costs. Considering the overheads coming from the implementation of QRAM and fault tolerance, the traveling salesperson problem seems to be one of the *least* likely candidates to achieve a practical quantum speedup when the nodes have large degree. For bounded degree graphs there is slightly more hope as quantum algorithms with close-to-quadratic speedups have been devised [789] that do not require QRAM.

Finally, let us mention quantum walk algorithms, which can also be viewed as a generalization of Grover search. However, quantum walks are more dis-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tant relatives of Grover search and can only be applied in more specific settings. They can be used for proving many nontrivial speedups in query complexity, however, the resulting algorithms are often not practical due to high space and/or gate complexity overheads, as is the case for the prototypical element distinctness problem. The query reduction is moderate $N \to N^{2/3}$ in the number of elements N, but the corresponding quantum algorithm [25] unfortunately uses a QRAM consisting of roughly $N^{2/3}$ registers; moreover, the QRAM must be able to store data in superposition.

There are nevertheless more practical quantum walk algorithms applicable, for example, to speed up backtracking algorithms [775, 27, 569, 743], which are among the most successful and widely used classical heuristics for solving SAT instances in practice. The quantum algorithm can achieve an essentially quadratic speedup compared to its classical backtracking variant. This approach is applicable to the traveling salesperson problem in the special case that the graph has degree at most 4 [789]. For resource estimates, see the earlier quoted reference [223]. A further extension of this algorithm is applicable to branch-and-bound algorithms [776, 247], and in some cases yields running times that are substantially better than what we know can be achieved by naively using Grover's algorithm. For example, it can find exact ground states for most instances of the Sherrington–Kirkpatrick model [933] in time $O(2^{0.226n})$ [776], which means about a quadratic speedup compared to classical methods. Branch-and-bound-based speedups can also be applied to solve mixed-integer programs, which include certain formulations of the portfolio optimization problem [247].

There is a plethora of other applications of quantum search speedups, ranging from machine learning [1040] to dynamic programming solutions of other NP-hard problems [28], which we do not discuss here for length constraints and due to discouraging resource estimates for (sub)quadratic quantum speedups.

4.2 Beyond quadratic speedups in exact combinatorial optimization

Overview

The discovery of Grover's algorithm [464] (later generalized to amplitude amplification) has long been the source of enthusiasm that quantum algorithms can be advantageous for combinatorial optimization, as it leads to quadratic asymptotic speedups for many concrete end-to-end search problems in this area. However, resource estimates indicate that early and intermediate-term

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

fault-tolerant devices will fail to deliver practical advantages when the available speedup is only quadratic, due to intrinsic overheads of quantum computation compared to classical computation (see, e.g., [223, 79]). Thus, identifying whether beyond-quadratic speedups are available is of principal importance for identifying end-to-end practical advantages in combinatorial optimization. Despite the fact that Grover's algorithm is optimal in the black-box (unstructured) setting, superquadratic speedups could be possible when the combinatorial optimization problem has a certain structure that can be better exploited by a quantum algorithm than a classical algorithm.

Unfortunately, many proposals that could conceivably deliver superquadratic speedups lack rigorous theoretical performance guarantees. This includes the quantum adiabatic algorithm and variational quantum algorithms such as the quantum approximate optimization algorithm (QAOA) [384], which is typically formulated to give approximate solutions, but at higher cost could also be used to find exact solutions. Limited analytic and numerical work provides some evidence (e.g., [179, 928]) that QAOA could outperform a vanilla application of Grover's algorithm to the k-SAT problem, but provides no definitive conclusion on the matter. Alternatively, a line of work in [505, 329] studies a different algorithm (related in certain aspects to the quantum adiabatic algorithm) and provides rigorous running time guarantees that *slightly* surpass Grover's algorithm.

However, while these algorithms may have a speedup over Grover's algorithm, this does not entail a superquadratic speedup over the *best* classical algorithm, which can often exploit structure in other ways to do much better than exhaustive search. Overall, it remains an open question whether quantum algorithms can provide superquadratic speedups for useful problems in exact combinatorial optimization.

Actual end-to-end problem(s) solved

Combinatorial optimization problems ask to find which solution is optimal among a finite set of possible candidates. Here, we stick to binary optimization on n bits, where the universe of possible candidates are bit strings $z = (z_1, z_2, \ldots, z_n) \in \{1, -1\}^n$. The input to the problem is a compact description of some cost function $C : \{1, -1\}^n \to \mathbb{R}$, and the desired output is the string z^* for which C is minimized. Let $E^* = C(z^*)$ denote the optimal value of the cost function. For simplicity we assume z^* is unique and E^* is known ahead of time.[3] This setting contrasts with that of *approximate* optimization, where the

[3] This assumption can often be relaxed at the expense of at most poly(n) overhead, for example, by iterating over all possible values E^* might take, which fall within a poly(n)-size range when the cost function consists of only poly(n) constant-size (integer-valued) terms.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

acceptable outputs include a much larger set of strings z that are not necessarily optimal solutions, but are still good enough, for example, because they achieve a nontrivial approximation ratio $|C(z)|/|E^*|$ with the optimal cost value. Classical and quantum algorithms for approximate optimization are often heuristic, making it more difficult to systematically study the complexity of the algorithms and the possibility that quantum algorithms may provide a speedup.

Concrete examples can be formed by choosing the function $C(z)$ to be a low-degree polynomial in the bits of z. For example, if C is a degree-2 polynomial in z, this is a quadratic unconstrained binary optimization (QUBO) problem, which is also equivalent to the classical Ising spin model from Eq. (1.3). If, furthermore, every term of C has degree exactly 2 (no degree-1 or constant terms) and every coefficient is either 0 or 1, then the problem is equivalent to a MAX-CUT problem. Finally, if C is a sum of degree-3 terms of the form

$$z_a z_b z_c + z_a z_b + z_a z_c + z_b z_c + z_a + z_b + z_c \,,$$

where

$$z_a, z_b, z_c \in \{z_1, -z_1, z_2, -z_2, \ldots, z_n, -z_n\} \,,$$

then the problem is equivalent to a MAX-3-SAT instance in conjunctive normal form. To see this, note that if $z_a = z_b = z_c = 1$, the term evaluates to 7, and for any other setting, it evaluates to -1. Thus, the solution z^* that optimizes C represents the bit string that minimizes the number of "unsatisfied" clauses for which $z_a = z_b = z_c = 1$. This is easily generalized from MAX-3-SAT to MAX-k-SAT.

For a fixed instance C, the quantum algorithms must find z^* with high probability over measurement outcomes. If it does so for every C chosen from some class of problem, we say it succeeds in the worst case. Alternatively, we can consider ensembles of instances chosen from some class of problem; if for a large fraction of instances from the ensemble, the algorithm finds z^* with high probability, then we say the algorithm succeeds in the average case.[4] A commonly considered average-case ensemble is the Sherrington–Kirkpatrick (SK) model [933], defined as

$$C(z) = \sum_{i=1}^{n} \sum_{j=i+1}^{n} J_{ij} z_i z_j \quad \text{where} \quad J_{ij} \sim \mathcal{N}(0,1), \tag{4.1}$$

[4] A more typical definition of the average-case complexity of an algorithm is the expected runtime required for it to find the solution z^*, averaged over both choice of instance and internal algorithmic randomness (i.e., classical coin flips or quantum measurement outcomes). This definition is related to the convention we follow, but it is more coarse grained as it does not distinguish between the two types of randomness, the latter of which can be boosted by repetition.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

where the coefficients J_{ij} are drawn randomly from a standard Gaussian distribution $\mathcal{N}(0,1)$. The SK model is relevant in spin glass theory, and can be generalized to higher-degree interactions, where it is referred to as the p-spin model [345]. Another ensemble is the random MAX-k-SAT ensemble, where MAX-k-SAT instances are generated by choosing each clause uniformly at random with some fixed clause-to-variable ratio (see, e.g., [307]).

Dominant resource cost/complexity

A vanilla application of Grover's algorithm to binary optimization problems achieves $O^*(2^{0.5n})$ running time, where notation $O^*(2^{an})$ is shorthand for $\text{poly}(n)2^{an}$. We cover three approaches to solving binary optimization problems on a quantum computer that have some potential to improve upon this running time. Note that all of these algorithms require polynomial (in fact, linear $O(n)$) space. However, their running time is expected to scale exponentially in n.

- First, we consider variational quantum algorithms, using the QAOA [384] as a representative. These algorithms are typically studied as efficient (polynomial-time) quantum algorithms that produce approximate solutions, that is, strings $z \neq z^*$ for which $C(z)$ is small, but not optimal. However, they may also be viewed as exact algorithms, since, if repeated a sufficient number of times, they eventually produce the exactly optimal z^*. The QAOA fixes a depth parameter p and variational parameters $\gamma = (\gamma_1, \ldots, \gamma_p)$ and $\beta = (\beta_1, \ldots, \beta_p)$ (sometimes these are set to some fixed instance-independent value, and sometimes they are variationally updated on subsequent repetitions of the algorithm). The QAOA starts in the n-qubit equal superposition state $|+\rangle^{\otimes n}$ and implements alternating rounds of rotations about the diagonal cost function C and a "mixing" operator $X = \sum_i X_i$, where X_i denotes the Pauli-X gate about qubit i. The state produced by QAOA is thus given by

$$|\psi_{\gamma,\beta}\rangle = \mathrm{e}^{-\mathrm{i}\beta_p X}\mathrm{e}^{-\mathrm{i}\gamma_p C}\cdots\mathrm{e}^{-\mathrm{i}\beta_2 X}\mathrm{e}^{-\mathrm{i}\gamma_2 C}\mathrm{e}^{-\mathrm{i}\beta_1 X}\mathrm{e}^{-\mathrm{i}\gamma_1 C}|+\rangle^{\otimes n}.$$

  If one makes a computational basis measurement of $|\psi_{\gamma,\beta}\rangle$, one obtains z^* with probability $|\langle z^*|\psi_{\gamma,\beta}\rangle|^2$. The expected number of repetitions required to obtain z^* is the inverse of this probability, and this running time can be quadratically sped up by performing amplitude amplification on top of the QAOA protocol; thus, the QAOA unitary is applied $O(|\langle z^*|\psi_{\gamma,\beta}\rangle|^{-1})$ times. Implementing the QAOA unitary typically requires only $p \cdot \text{poly}(n)$ gates, as each of the rotations about X and C are efficient to implement. For hard combinatorial optimization problems such as typical MAX-k-SAT instances, the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

expectation is that the total running time required will be exponential. If the depth p is chosen to be constant or even poly(n), the dominant cost will come from the $O(|\langle z^*|\psi_{\gamma,\beta}\rangle|^{-1})$ repetitions required to amplify the $|z^*\rangle$ state. Alternatively, one can reduce the number of repetitions needed to $O(1)$ at the expense of taking p to be very large (at least exponentially large in n); indeed, for sufficiently large p, the QAOA can be viewed as a Trotterized simulation of the adiabatic algorithm [384].

There is some analytic evidence that the QAOA may outperform Grover's algorithm at finding the exact solution for constant p in certain cases. Reference [179] studied the QAOA applied to hard (i.e., near the satisfiability threshold) k-SAT instances with instance-independent choice of γ, β for constant p, and developed an analytic formula for the expected success probability $|\langle z^*|\psi_{\gamma,\beta}\rangle|^2$ averaged over random instance in the limit $n \to \infty$. This formula was evaluated numerically and suggested, for example, that the average success probability behaves as $2^{-0.33n}$ for $p = 10$ on 8-SAT. One might be tempted to declare that this implies an overall average running time of $O^*(2^{0.33n/2})$, substantially better than Grover, but such a conclusion is not analytically supported as the average of the inverse probability can be much larger than the inverse of the average probability. Nevertheless, it provides intriguing evidence in favor of such a conclusion. Further numerical evidence that QAOA may be effective as an exact algorithm was provided in [928], which numerically assessed the performance of QAOA on instances of the low autocorrelation binary sequences (LABS) problem up to $n = 40$, although compared to the best classical heuristic solver, the advantage appeared to be subquadratic.

- Second, we consider the quantum adiabatic algorithm [382, 16]. The standard approach, as applied to binary optimization problems, is to start in the state $|+\rangle^{\otimes n}$ and evolve by a Hamiltonian that interpolates along a path $H(s)$ parameterized by $s \in [0, 1]$, given by

$$H(s) = (1 - s)(-X) + sC\,. \tag{4.2}$$

It is important to note that the ground state of $H(0)$ is $|+\rangle^{\otimes n}$ and the ground state of $H(1)$ is $|z^*\rangle$. This evolution can be simulated on a fault-tolerant gate-based quantum computer using Hamiltonian simulation, and its running time is dominated by the inverse of the minimum spectral gap $\Delta_{\min}$ of $H(s)$. That is, the gate complexity to run the algorithm and produce $|z^*\rangle$ scales as at least $\Delta_{\min}^{-1}$ and possibly a larger power of $\Delta_{\min}^{-1}$. Much numerical work has been done on the performance of the adiabatic algorithm on small instances of combinatorial optimization problems, but it generally lacks analytical guarantees. The expectation is that $\Delta_{\min}$ will be exponentially small

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[638, 1071, 517] in n (or worse, see, e.g., [23, 1032]), meaning the running time of the algorithm is exponentially large, but it remains possible that it surpasses the $O^*(2^{0.5n})$ running time of Grover's algorithm in some cases, and could in principle deliver a superquadratic speedup.

- Third, we consider the *short-path* algorithm studied in [505, 506, 508] and a dual version of the algorithm studied in [329]. The goal of these algorithms was to be able to provide a rigorous guarantee that the algorithm can find z^* in time $2^{(0.5-c)n}$ for some value of $c > 0$. Similar to the adiabatic algorithm, the short-path algorithm also considers a single-parameter family of Hamiltonians

$$H(s) = (1-s)f_X\left(-\frac{X}{n}\right) + sf_Z\left(\frac{C}{|E^*|}\right), \tag{4.3}$$

where $f_X, f_Z : \mathbb{R} \to \mathbb{R}$ are monotonic filter functions, and each term X/n and $C/|E^*|$ are normalized to have minimum value -1. The idea of the short-path algorithm is to, rather than evolve smoothly from $s = 0$ to $s = 1$, perform a pair of discrete "jumps." The first jump goes from the ground state $|+\rangle^{\otimes n}$ at $s = 0$ to the ground state $|\psi_b\rangle$ of an intermediate point with $s = b$. The second jump goes from $|\psi_b\rangle$ to the ground state $|z^*\rangle$ at $s = 1$. The jumps are accomplished with quantum phase estimation (or more advanced versions utilizing the quantum singular value transformation) of the Hamiltonian H_b combined with amplitude amplification. The running time of the algorithm is [329, Theorem 1]

$$\text{poly}(n) \cdot \frac{1}{\Delta} \cdot \left(\frac{1}{|\langle +|\psi_b\rangle|} + \frac{1}{|\langle \psi_b|z^*\rangle|}\right), \tag{4.4}$$

where Δ is the spectral gap of the Hamiltonian $H(b)$. The Δ^{-1} factor comes from the need to perform phase estimation at $O(\Delta)$ resolution to successfully prepare $|\psi_b\rangle$, and the two additive inverse overlap terms represent the number of rounds of amplitude amplification for the first and second jumps, respectively. In [505], filter functions $f_X(x) = x^K$ for odd integers K (e.g., $K = 3$) and $f_Z(x) = x$ were chosen, and b was chosen close to 1, such that the first term of Eq. (4.3) could be viewed as a small perturbation of the second term. If C is an instance of MAX-Ek-LIN2, that is, if it is a polynomial for which all monomials are degree exactly k, then it was shown that certain conditions on the spectral density of C near the optimal cost value imply sufficient analytic control of Δ and the other parameters in Eq. (4.4) such that the algorithm runs in time $O^*(2^{(0.5-c)n})$ for $c > 0$. However, it remained unclear when these conditions were met. Inspired by [505], [329] proposed using the filter functions $f_X(x) = x$ and $f_Z(x) = \min(0, (x + 1 - \eta)/\eta)$ for a fixed choice of $\eta \in [0, 1]$, and chose a value of s close to 0 (rather than

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

close to 1). In this sense, the algorithm in [329] is dual to that of [505]. These modifications allowed additional statements to be proved. For example, it was unconditionally shown that the algorithm solves k-SAT (whether or not a formula has a fully satisfiable solution) in time upper bounded by $O^*(2^{(0.5-c)n})$ for a (extremely small) constant $c > 0$, and that the same is true for typical instances of the SK model and its higher-body generalization (p-spin model), a polynomial speedup over Grover's algorithm and superquadratic advantage over classical exhaustive search.

Existing resource estimates

Reference [895] compiled resource estimates for various primitive tasks related to combinatorial optimization. For example, it estimated that for an $n = 512$ instance of the SK model, implementing a single QAOA step $e^{-i\beta_j X} e^{-i\gamma_j C}$ would require 577 logical qubits and 5.0×10^5 Toffoli gates. A similar estimate would hold for performing a single step of adiabatic evolution with a first-order product formula. The total logical estimate for finding z^* would be the product of the depth of the circuit and any number of repetitions or rounds of amplitude amplification. An estimate of the physical resource cost could then be computed for a specific fault-tolerant architecture. Without knowing the number of repetitions, it is hard to give precise estimates, but a rough attempt was made in [79] for different speedup factors. There, under different possible assumptions on the amount of classical parallelism available, a breakeven point was estimated for different possible polynomial speedups (quadratic, cubic, and quartic). It was found that with a quartic speedup, the breakeven point could be reasonable (on the order of seconds to hours) even assuming the availability of classical parallelism.

Caveats

There are several caveats. The most salient one is that for most of the algorithms above, there is no provable beyond-Grover advantage. Meanwhile, in the case of [329], the size of the provable beyond-Grover advantage is miniscule. The prospect of these algorithms is thus left to extrapolations from numerical simulations carried out at very small instance sizes and speculation based on physical principles.

A second important caveat is that to deliver practical superquadratic speedups, the performance of the quantum algorithm needs to be compared to the best classical algorithm, which is often substantially better than the $O^*(2^n)$ running time of exhaustive enumeration. For example, 3-SAT problems are classically solvable in $O^*(2^{0.39n})$ time [497].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Along these lines, a third caveat is the existence of classical "quantum Monte Carlo" algorithms (see, e.g., [383, 188, 570, 322, 324]), which can, under certain conditions, classically simulate the quantum algorithms described above. This is because the Hamiltonians in Eqs. (4.2) and (4.3) are *stoquastic* Hamiltonians, defined by the property that their off-diagonal matrix elements are non-positive (when written in the computational basis). Stoquasticity implies that the ground state of the Hamiltonian can be written such that all amplitudes are non-negative real numbers [191], meaning that these Hamiltonians avoid the so-called "sign problem" enabling the potential application of quantum Monte Carlo techniques. To be clear, it remains possible that quantum algorithms for these combinatorial optimization problems involving stoquastic Hamiltonians can evade classical simulation—indeed, superpolynomial oracle separations have been shown between classical computation and adiabatic quantum computation restricted to stoquastic paths [510, 432]—but it is something to keep in mind when designing algorithms based on stoquastic Hamiltonians.

A final caveat is that the quantum algorithms described here are typically not amenable to parallelization, although in principle QAOA could be parallelized if one opts not to use amplitude amplification (resulting in worse asymptotic complexity). This lies in stark contrast to many classical optimization algorithms for exact combinatorial optimization which are highly parallelizable, a feature that can be exploited to significantly reduce the running time of these classical algorithms on high-performance computers, making achieving practical quantum advantage more difficult [79].

Comparable classical complexity and challenging instance sizes

For many binary optimization problems, there exist classical algorithms that exploit the structure of the problem to perform significantly better than exhaustive search. For example, the best 3-SAT algorithm runs in time $O^*(2^{0.39n})$ and in general k-SAT can be solved in time $2^{(1-\Omega(1/k))n}$ [497]. This running time suggests the solution will be impractical once n is on the order of 100. The algorithm analyzed in [497] is designed for the worst case, and it is likely not the best practical algorithm for typical instances. For random instances, the hardness of k-SAT depends sensitively on the clause-to-variable ratio α. Remarkably, heuristic algorithms can succeed at finding a satisfiable solution for typical instances with thousands or even tens of thousands of variables even very close to the satisfiability threshold α_c where most instances become unsatisfiable (e.g., [739]). However, these algorithms are expected to fail sufficiently close to the satisfiability threshold and in the worst case.

Similarly, the SK model admits a classical branch-and-bound algorithm guaranteed to run in time $2^{0.45n}$ (for a large fraction of instances) and likely

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

better than that in practice [776]. However, once the interaction degree becomes larger than 2, the problem becomes significantly harder. The branch-and-bound algorithm is not known to generalize to the p-spin model, and for $p \geq 3$ there is no known classical algorithm that provably achieves $2^{(1-c)n}$ for any constant c (although it has not garnered much attention, see [329]). Similarly, in contrast to k-SAT, the MAX-k-SAT problem (i.e., the version of the problem that asks for the optimal assignment even if it does not satisfy all the clauses) only has a $O^*(2^{(1-c)n})$ time algorithm for $k = 2$, and, notably, this algorithm requires exponential space [1047].

Speedup

As there are generally no rigorous running time guarantees for the quantum algorithms, the speedup cannot be estimated. However, it is worth emphasizing that for hard combinatorial optimization problems, the speedup could be superquadratic, but it is not expected to be superpolynomial.

The rigorous results of [329] establish a beyond-Grover running time, but the only case in which the speedup is beyond quadratic when compared with the best known classical algorithm is the p-spin model with $p \geq 3$ (here, the comparison benefits from little work on classical algorithms for the problem).

We also mention the result of [904], which studies a quantum algorithm for random instances of a QUBO-like combinatorial optimization problem with a "planted" optimal solution—the goal is to exactly or approximately find the planted solution, or alternatively to simply distinguish instances drawn from the ensemble with planted solutions from instances drawn from the ensemble without a planted solution. The algorithm generalizes the tensor PCA algorithm of [511] and gives a *quartic* speedup over its closest classical counterpart, although it is unclear if this speedup can extend to non-planted scenarios as well.

NISQ implementation

The QAOA approach is amenable to NISQ implementation (assuming one opts not to apply amplitude amplification on top of it), since the quantum circuit one needs to implement is fairly shallow depth. In this case, the effect of uncorrected errors in the NISQ device may degrade the performance (and require more repetitions to extract the optimal bit string z^*). Similarly, on a NISQ quantum annealer [591, 16], one could run a noisy version of the quantum adiabatic algorithm and repeat until finding the optimal bit string z^*.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Outlook

In contrast to algorithms for approximate optimization, which are often heuristic but run in polynomial time, algorithms for exact optimization are often more rigorous but run in exponential time. For quantum computers to be impactful for exact combinatorial optimization, we require great advancements in the estimated clock speeds of quantum hardware and the overheads of fault-tolerant quantum computing, or else the development of quantum algorithms that significantly improve upon existing (sub)quadratic Grover-type speedups—either quantitatively (bigger speedups) or qualitatively (e.g., requiring only shallow circuits). Although ideas have been proposed that could potentially deliver such improvements, they either come without provable guarantees, provide only minor superquadratic improvement, or only apply to artificial problems. Much more attention shall be devoted to studying these quantum algorithms and developing new ones if we are to leverage them into actual practical advantages, especially considering the extensive amount of work devoted to developing sophisticated classical algorithms for these problems.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

5

Continuous optimization

Continuous optimization problems arise throughout science and industry. On their face, continuous optimization problems rarely seem quantum mechanical; nevertheless, quantum algorithms have been proposed for accelerating both convex and nonconvex continuous optimization. Most of the research on these algorithms thus far has been to develop and utilize the diverse set of primitive ingredients that give rise to potential quantum advantage in this space, without an eye toward the end-to-end practicality of the algorithms. Developing a better understanding of the practicality of these approaches should be a focus of future work.

We refer the reader to [6] for a comprehensive survey of quantum methods for continuous and combinatorial optimization.

The authors are grateful to Sander Gribling for reviewing this chapter.

5.1 Zero-sum games: Computing Nash equilibria

Overview

In a two-player zero-sum game, each player independently chooses an action and then receives a "payoff" (such that the sum of the payoffs is always zero) that depends on which pair of actions was chosen. A Nash equilibrium is an optimal way of (probabilistically) choosing an action that maximizes a player's worst-case payoff. The problem of computing a Nash equilibrium is, in a certain sense, equivalent to solving a linear program (LP): computing a Nash equilibrium is a special case of LP, and conversely any LP can be reduced to computing a Nash equilibrium at the expense of introducing dependencies on a certain instance-specific "scale-invariant" precision parameter [46]. How-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ever, the quantum approach to solving LPs based on the multiplicative weights update method [46] is more efficient in the special case of computing Nash equilibria, and has fewer caveats—notably, it avoids the dependence on the difficult-to-predict scale-invariant precision parameter. It gives a potentially quadratic speedup over its classical counterpart.

Actual end-to-end problem(s) solved

A two-player zero-sum game is defined by an $n \times m$ matrix A called the "payoff matrix," which specifies how much player 1 wins from player 2 when player 1 chooses action $i \in [n]$ and player 2 chooses action $j \in [m]$. A pure strategy is one in which the players deterministically choose one fixed action $i \in [n]$ (or $j \in [m]$) in each game. By contrast, a mixed strategy is one in which players randomly choose an action, according to some probability distribution. Assume the entries of A are between -1 and 1. A Nash equilibrium is an optimal (generally mixed) strategy that maximizes a player's worst-case payoff regardless of the other player's choice. That is, a distribution $y \in \Delta^m$, where Δ^m denotes the m-dimensional probability simplex, is an optimal strategy for player 2 if it is the argument that optimizes the equation

$$\lambda^* = \min_{y \in \Delta^m} \max_{i \in [n]} e_i^\mathsf{T} A y,$$

where $[n]$ denotes the set of actions available to player 1, and e_i denotes a basis state associated with action i. The quantity λ^* is the value of the game. This can be rewritten explicitly [46] as the following LP

$$\min_{y \in \mathbb{R}^m} \lambda$$
$$\text{subject to} \qquad Ay \leq \lambda \mathbf{1}, \qquad \sum_j y_j = 1, \qquad y_j \geq 0 \ \forall j,$$

where $\mathbf{1}$ is the all-ones vector. The dual LP for the above then corresponds to computing the Nash equilibrium for player 1.

The end-to-end problem solved is to, given access to the entries of the matrix A and an error parameter ϵ, compute a probability vector y such that

$$Ay \leq (\lambda^* + \epsilon)\mathbf{1}\,.$$

Dominant resource cost/complexity

The quantum algorithm builds on a classical algorithm based on the multiplicative weights update method from [459]. With probability at least $1 - \delta$, the classical algorithm finds a solution y that approximates a Nash equilibrium to error ϵ after $\lceil 16 \ln(nm/\delta)/\epsilon^2 \rceil$ iterations, where the cost per iteration is $n + m$ queries to the entries of the matrix A and $O(n + m)$ other arithmetic operations

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[46, Lemma 3]. An important subroutine of each iteration is a Gibbs sampling step for a diagonal matrix (a special case of the general quantum Gibbs sampling problem in which any Hermitian matrix is allowable). When the matrix A is sparse, the number of queries per iteration can be reduced to $2s$, where s is the maximum number of nonzero entries in a row or column of A, and the total time per iteration can be reduced to $\widetilde{O}(s)$ [46, Lemma 4].

The quantum algorithm assumes coherent access to the matrix entries of A. Through amplitude amplification and the related subroutines of amplitude estimation and minimum finding, the quantum algorithm of [46] speeds up the Gibbs sampling task and reduces the maximum cost of an iteration to $\widetilde{O}(\sqrt{n+m}/\epsilon)$ queries to the matrix elements of A and an equal amount of time complexity, where $\widetilde{O}$ notation suppresses logarithmic factors. In the case that the matrices are sparse, the maximum cost of an iteration is reduced to $\widetilde{O}(\sqrt{s}/\epsilon^{1.5})$. The work of [178] introduces a technique called dynamic Gibbs sampling, which exploits the fact that the distribution to be sampled changes slowly from iteration to iteration and further reduces the iteration cost to $\widetilde{O}(\sqrt{n+m}/\epsilon^{1/2} + 1/\epsilon)$ in the dense case. This gives a total query and time complexity roughly given by

$$\text{dense:} \quad \left(\frac{16\ln(nm)}{\epsilon^2}\ \text{iters.}\right) \times \left(\widetilde{O}\left(\frac{\sqrt{n+m}}{\sqrt{\epsilon}} + \frac{1}{\epsilon}\right)\ \text{per iter.}\right) = \widetilde{O}\left(\frac{\sqrt{n+m}}{\epsilon^{2.5}} + \frac{1}{\epsilon^3}\right)$$

$$\text{sparse:} \quad \left(\frac{16\ln(nm)}{\epsilon^2}\ \text{iters.}\right) \times \left(\widetilde{O}\left(\frac{\sqrt{s}}{\epsilon^{1.5}}\right)\ \text{per iter.}\right) = \widetilde{O}\left(\frac{\sqrt{s}}{\epsilon^{3.5}}\right).$$

This complexity assumes access to a quantum random access memory (QRAM). Without a QRAM, the cost per iteration increases by a factor $\widetilde{O}(1/\epsilon^2)$.

See also [680], which independently from [46] gave a quantum algorithm that solves zero-sum games with slightly worse ϵ dependence, as well as [681], which gave quantum algorithms for generalizations of zero-sum games to other vector norms.

Existing resource estimates

There are no existing explicit resource estimates for this algorithm.

Caveats

- Due to poor dependence of the complexity on the error ϵ, this algorithm is only likely to be useful in situations where it is not necessary to learn the optimal strategy to high precision. It is unclear when such situations arise in practice.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- As mentioned above, if no QRAM is available, the runtime suffers a $\widetilde{O}(1/\epsilon^2)$ time slowdown.
- A fully end-to-end analysis should also consider the exact way that the queries to the matrix entries of A are implemented. If they are given in a classical database, a large $O(nm)$-size QRAM may also be required to implement the queries in polylog(mn) time. Note that this would be separate from the $\widetilde{O}(1/\epsilon^2)$-size QRAM the algorithm uses to reduce the time complexity. To avoid the QRAM requirement for implementing a query, it must be the case that the matrix entries are efficiently computable in some other way.

Comparable classical complexity and challenging instance sizes

The classical version of the quantum algorithm has time and query complexity given by [46, Section 2]

$$\text{dense:} \quad \left(\frac{16\ln(nm)}{\epsilon^2} \text{ iters.}\right) \times (O(n+m) \text{ per iter.}) = \widetilde{O}\left(\frac{n+m}{\epsilon^2}\right)$$

$$\text{sparse:} \quad \left(\frac{16\ln(nm)}{\epsilon^2} \text{ iters.}\right) \times \left(\widetilde{O}(s) \text{ per iter.}\right) = \widetilde{O}\left(\frac{s}{\epsilon^2}\right).$$

Alternatively, the problem could be solved using other approaches for solving the associated LP. Classical interior point methods for LPs can achieve $O(n^\omega \log(1/\epsilon))$ runtime in the common case that $m = O(n)$ [304], where $\omega < 2.37$ is the matrix multiplication exponent. This runtime exhibits better ϵ dependence at the expense of worse n dependence. Note that quantum interior point methods have also been proposed for conic programs like LPs, but whether they could yield a speedup over classical interior point methods would depend on the scaling of certain instance-specific parameters.

Speedup

The quantum complexity has a quadratic improvement in complexity with respect to the parameter $n + m$, and a polynomial slowdown with respect to the parameter ϵ.

Outlook

It is difficult to assess whether a practical advantage could be obtained in the setting of zero-sum games without further investigation of how queries to matrix elements are accomplished, an assessment of constant prefactors involved in the algorithm, and consideration of any additional overheads from fault-tolerant quantum computation. The theoretical speedup available is quadratic

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

and may require a medium- or large-scale QRAM. This speedup may not be sufficiently large to overcome these overheads in practice.

It is perhaps instructive to compare the outlook of zero-sum games to conic programming more generally. On the one hand, unlike the algorithm for general SDPs and LPs, the algorithm for zero-sum games does not have a complexity dependence on instance-specific parameters denoting the size of the primal and dual solutions. This makes it easier to evaluate the runtime of the algorithm and more likely that it can be an effective algorithm. On the other hand, a core subroutine of the quantum algorithm is to perform *classical* Gibbs sampling quadratically faster than a classical computer can using techniques like amplitude amplification. However, it is not clear how the speedup could be made greater than quadratic, even in special cases. A similar subroutine is required in the multiplicative weights approach to solving SDPs, but in that case, the Gibbs state to be sampled is a truly quantum state (i.e., nondiagonal in the computational basis), rather than a classical state. Using more advanced methods for Gibbs sampling, it is possible that in some special cases there could be a superquadratic quantum speedup for SDPs that would not be available for the simpler case of LPs and zero-sum games.

5.2 Conic programming: Solving LPs, SOCPs, and SDPs

Overview

Conic programs are a specific subclass of convex optimization problems, where the objective function is linear and the convex constraints are restrictions to the intersection of affine spaces and certain cones within $\mathbb{R}^n$. Commonly considered cones are the positive orthant, the second-order cone ("ice-cream cone"), and the semidefinite cone, which give rise to linear programs (LPs), second-order cone programs (SOCPs), and semidefinite programs (SDPs), respectively. This framework remains quite general, and many real-world problems can be reduced to a conic program. However, the additional structure of the program allows for more efficient classical and quantum algorithms, compared to completely general convex problems.

Algorithms for LPs, SOCPs, and SDPs have long been a topic of study. Today, the best classical algorithms are based on interior point methods (IPMs) [304, 778, 538] and cutting-plane methods [671, 575], but other algorithms based on the multiplicative weights update (MWU) method [57, 56, 58] exist and can be superior in a regime where high precision is not required. Both of these approaches can be turned into quantum algorithms with potential to deliver asymptotic quantum speedup for general LPs, SOCPs, and SDPs. How-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ever, the runtime of the quantum algorithm typically depends on additional instance-specific parameters, which makes it difficult to produce a general apples-to-apples comparison with classical algorithms.

Actual end-to-end problem(s) solved

- Linear programs (LPs) are the simplest convex program. An LP instance is specified by an $m \times n$ matrix A, an n-dimensional vector c, and an m-dimensional vector b. The problem can then be written as

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \langle c, x \rangle \\ &\text{subject to } Ax = b \\ &\qquad x_i \geq 0 \text{ for } i = 1, \ldots, n\,, \end{aligned}$$

  where notation $\langle u, v \rangle$ denotes the standard dot product of vectors u and v. The function $\langle c, x \rangle$, which is linear in x, is called the objective function, and a point x is called feasible if it satisfies the linear equality[1] constraints $Ax = b$ as well as the positivity constraints $x_i \geq 0$ for all i. We denote the feasible point that optimizes the objective function by x^*. Let ϵ be a precision parameter. The actual end-to-end problem solved is to take as input a classical description of the problem instance (c, A, b, ϵ) and output a classical description of a feasible point x for which $\langle c, x \rangle \leq \langle c, x^* \rangle + \epsilon$. The set of points that obey the positivity constraints $x_i \geq 0$ forms the positive orthant of the vector space $\mathbb{R}^n$. This set meets the mathematical definition of a convex cone: for any points u and v in the set and any non-negative scalars $\alpha, \beta \geq 0$, the point $\alpha u + \beta v$ is also in the set.
- Second-order cone programs (SOCPs) are formed by replacing the positivity constraints in the definition of LPs with one or more second-order cone constraints, where the second-order cone of dimension k is defined to include points $(x_0; x_1; \ldots; x_{k-1}) \in \mathbb{R}^k$ for which $x_0^2 \geq x_1^2 + \cdots + x_{k-1}^2$.
- Semidefinite programs (SDPs) are formed by replacing the n-dimensional vector x in the definition of LPs with an $n \times n$ symmetric matrix X and replacing the positive orthant constraint with the conic constraint that X is a positive semidefinite matrix. Denote the set of $n \times n$ symmetric matrices by $\mathbb{S}^n$, and for any pair of matrices $U, V \in \mathbb{S}^n$, define the notation $\langle U, V \rangle = \mathrm{tr}(UV)$ (which generalizes the standard dot product). Then, an SDP instance is specified by matrices $C, A^{(1)}, A^{(2)}, \ldots, A^{(m)} \in \mathbb{S}^n$, as well as $b \in \mathbb{R}^m$, and

[1] Inequality constraints of the form $Ax \leq b$ can be converted to linear equality constraints and positivity constraints by introducing a vector of slack variables s and imposing $Ax + s = b$ and $s_i \geq 0$ for all i. An analogous trick is possible for SOCP and SDP.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

can be written as

$$\begin{aligned} &\min_{X \in \mathbb{S}^n} \langle C, X \rangle \\ &\text{subject to } \langle A^{(j)}, X \rangle = b_j \text{ for } j = 1, \ldots, m \\ &X \succeq 0\,, \end{aligned}$$

where $X \succeq 0$ denotes the constraint that X is positive semidefinite.

In the LP or SDP case, we might also require as input parameters R and r, where R is a known upper bound on the size of the solution in the sense that $\sum_i |x_i| \leq R$ (LP) or $\mathrm{tr}(X) \leq R$ (SDP), and where r is an analogous upper bound on the size of the solution to the *dual* program (not written explicitly here, see [181, 48, 45]).

Dominant resource cost/complexity

Two separate approaches to solving conic programs with quantum algorithms have been proposed in the literature. Both methods start with classical algorithms and replace some of the subroutines with quantum algorithms.

(i) Quantum interior point methods (QIPMs) for LPs [610], SOCPs [612, 68], and SDPs [610, 70, 537] have been proposed. In the standard approach, these methods start with classical interior point methods, for which the core step is solving a linear system, and simply replace the classical linear system solver with a quantum linear system solver (QLSS), combined with pure state quantum tomography. Given a linear system $Gu = v$, the QLSS produces a quantum state $|u\rangle$, and quantum tomography is subsequently used to gain a classical estimate of the amplitudes of $|u\rangle$ in the computational basis. The QLSS ingredient introduces complexity dependence on a parameter $\kappa = \|G\| \|G^{-1}\|$, the condition number of G, where $\|\cdot\|$ denotes the spectral norm. Additionally, the QLSS requires that the classical data defining G be loaded in the form of a block-encoding, for which the standard construction introduces a dependence on the factor $\zeta = \|G\|_F \|G\|^{-1}$, where $\|\cdot\|_F$ denotes the Frobenius norm. Finally, the tomography ingredient introduces a complexity dependence on a parameter ξ, defined as the precision to which the vector u must be classically learned, measured in ℓ_2 norm. Assuming m is on the order of the number of degrees of freedom (i.e., $O(n)$ in the case of LP and SOCP, and $O(n^2)$ in the case of SDP), the number of queries the QIPM makes to block-encodings of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the input matrices is

$$\text{LP, SOCP [68]:} \qquad \widetilde{O}\left(\frac{n^{1.5}\zeta\kappa}{\xi}\log(1/\epsilon)\right)$$

$$\text{SDP [610, 70]:} \qquad \widetilde{O}\left(\frac{n^{2.5}\zeta\kappa}{\xi}\log(1/\epsilon)\right),$$

where the $\widetilde{O}$ notation hides logarithmic factors. Note that depending on how ξ is defined, extra factors of κ may be required. Moreover, note that the complexity statements in [70] go further and analyze the worst-case dependence of ξ on the overall error ϵ, and additionally make the worst-case replacement $\zeta \leq O(n)$—this explains the deviation in our presentation from the bounds in [70]. We do not include these worst-case assumptions on ζ and ξ because it is possible they are not achieved in practice.[2] Generally speaking, the numerical values of κ, ζ, and ξ are not possible to determine in advance for a specific application; empirical investigations at small system sizes such as those of [611, 328] require an assumption that trends observed accurately extrapolate to other untested instances and to larger system sizes. The block-encoding queries can be executed in circuit depth polylog$(n+m, 1/\epsilon)$, which can also be absorbed into the $\widetilde{O}$ notation (although it is important to note that the circuit *size* is generally $O(n^2)$)—this is equivalent to an assumption of log-depth quantum random access memory (QRAM). If the input matrices are sparse or given in a form other than as a list of matrix entries, there may be other more efficient methods for block-encoding; in this case the parameter ζ might be replaced with another parameter $\alpha > 1$, whose value would depend on the block-encoding method. It should also be noted that in addition to the quantum complexity quoted above, the QIPM can require (depending on the precise formulation of the QIPM) purely classical complexity on the order of $O(n^{2.5})$ for LP/SOCP and $O(n^{4.5})$ for SDP.

Alternatives to the standard approach above have been proposed. For "tall" LPs (where $m \gg n$), one can quantize interior point methods in a distinct way that avoids the QLSS and dependence on any condition numbers. Specifically, [51] gave an algorithm that runs in time $\widetilde{O}(\sqrt{m}\log(1/\epsilon)) \cdot \text{poly}(n)$. The algorithm leverages primitives for spectral approximation (i.e., given a tall matrix B, finding a smaller matrix

[2] For example, numerical results in [611] for small instances of an SOCP formulation of the portfolio optimization problem suggested that the ζ parameter was upper bounded by a small constant, and similar numerical investigations in [328] suggest that ξ can be independent of the target error ϵ, at least for the instances that were simulated.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$\tilde{B}$ for which $\tilde{B}^\intercal \tilde{B} \approx B^\intercal B$), and approximate matrix-vector multiplication, as well as multivariate mean estimation [310] (which is related to quantum gradient estimation).

(ii) Quantum algorithms based on the multiplicative weights update (MWU) method have been proposed for SDP [181, 182, 48, 45] and LP [48, 46]. The quantum algorithm closely follows the classical algorithm based on MWU to iteratively update a candidate solution to the program. Each iteration is carried out using quantum subroutines, including Gibbs sampling, as well as Grover search and quantum minimum finding [367, 48] (a direct application of Grover search). Let s denote the sparsity, that is, the maximum number of nonzero entries in any row or column of the matrices composing the problem input (thus, $s \leq \max(m, n)$). Then, the number of queries the algorithm makes to the matrix entries (assuming a sparse access input model) has been upper bounded by

$$\text{LP [178]:} \qquad \widetilde{O}\left(\sqrt{s}\left(\frac{rR}{\epsilon}\right)^{3.5}\right)$$

$$\text{SDP [45]:} \qquad \widetilde{O}\left(s\sqrt{m}\left(\frac{rR}{\epsilon}\right)^{4} + s\sqrt{n}\left(\frac{rR}{\epsilon}\right)^{5}\right),$$

where r, R are the parameters related to the size of the primal and dual solutions, defined above. The sparse access queries can be implemented with quantum circuits of size polylog(m, n) if the matrix entries are given by succinct formulas computable in polylog(n, m) time. Otherwise, their implementation can be accomplished with circuits of polylog(m, n) depth (but poly(m, n) size) assuming availability of log-depth QRAM. In [45], the input model was generalized to a "quantum operator input model," based on block-encodings where s is replaced by the block-encoding normalization factor α in the runtime expressions, but here again the full end-to-end complexity must account for the gate cost of implementing the block-encoding. Note that it is possible the ϵ-dependence of the runtime for LP could be slightly improved by applying the dynamic Gibbs sampling method of [178] together with the reduction from LP to zero-sum games in [46].

The runtime expressions for the QIPM approach and the MWU approach are not directly comparable, as the former depends on instance-specific parameters κ, ζ, and ξ, while the latter depends on instance-specific parameters r and R. However, note that the explicit n-dependence is better in the case of MWU than QIPM, while the ϵ-dependence is worse.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Existing resource estimates

Neither of the approaches for conic programs have garnered study at the level of resource estimates for physical devices. Reference [328] performed a resource analysis for a QIPM at the logical level, but did not analyze additional overheads due to error correction. The goal of that analysis was to completely compile the QIPM for SOCP into Clifford gates and T gates, and then to numerically estimate the parameters κ, ζ, and ξ for the particular use case of financial portfolio optimization, which can be reduced to SOCP. A salient feature of the QIPM is that $O(n+m) \times O(n+m)$ matrices of classical data must be repeatedly accessed by the QLSS via block-encoding, necessitating a large-scale QRAM with $O(n^2)$ qubits. Accordingly, for SOCPs with $n = 500$ and $m = 400$ (which are still easily solved on classical computers) it was estimated that 8 million logical qubits would be needed. The total number of T gates needed for the same instance size was on the order of 10^{29}, which can be distributed over roughly 10^{24} layers. These estimates would likely be improved by incorporating subsequent improvements to the underlying primitives of tomography [49] and QLSS [571, 327].

We are not aware of an analogous logical resource analysis for the MWU approach to conic programming. Such an analysis would be valuable and should ideally choose a specific use case to be able to evaluate the size of all parameters involved. A use case that may fit this criteria is solving the SDP relaxation of binary quadratic optimization problems, where r and R can be bounded; quantum algorithms for this task have been studied in [183, 71].

Caveats

- The QIPM approach requires a large-scale QRAM of size $O(n^2)$. This is a necessary ingredient to retain any hope of a speedup, and for relevant choice of n the associated hardware requirements could be prohibitively large. Note that recent work of [69] gave a method for solving LPs that, like QIPMs, follows the central path to the optimal point, but does so in a distinct, non-iterative way, and has the potential for a small polynomial speedup without the need for a QRAM.
- The standard QIPM approach has a weak case for a large asymptotic speedup: even under optimal circumstances, the asymptotic speedup over classical interior point methods is less than quadratic. See Chapter 22 on the QIPM approach for more information.
- The MWU approach also requires a large-scale QRAM of size $O(\min(ms, ns))$ to implement the queries to the arbitrary entries of the s-sparse input matrices. This could be avoided if the matrix elements are efficiently computable.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- Beyond the number of queries to the input data (and the cost to implement those queries), the MWU approach for LP and SDP requires some additional complexity deriving from a step in the algorithm where it prepares a state with $O(R^2r^2/\epsilon^2)$ nonzero amplitudes stored in a classical database. The cost of state preparation from classical data is $\widetilde{O}(R^2r^2/\epsilon^2)$ total gates, which can be parallelized to circuit depth $\text{polylog}(R^2r^2/\epsilon^2)$. If the cost of state preparation is taken to be equal to the circuit depth (similar to the assumption of access to a medium-scale QRAM), the additional complexity is on the same order as the number of queries. If the cost is taken to be equal to the circuit size, the additional complexity is a factor $\widetilde{O}(R^2r^2/\epsilon^2)$ larger than the query complexity quoted above.
- The MWU approach has poor dependence on error ϵ; for SDPs it is ϵ^{-5}. Even at modest choices of ϵ, this may lead the algorithm to be impractical pending significant improvements.
- A general caveat that applies to both approaches is that the appearance of instance-specific parameters makes it difficult to predict the performance of these algorithms for more specific applications.

Comparable classical complexity and challenging instance sizes

As in the quantum case, there are multiple distinct approaches in the classical case.

(i) Classical interior point methods (CIPMs): There exist fast IPM-based software implementations for solving conic programs, such as ECOS [355], MOSEK [38], Gurobi [470], SCIP [164], and CPLEX.[3] These solvers can solve instances with thousands of variables in a matter of seconds on a standard laptop (e.g., [355]). However, the runtime scaling is poor and scaling too far beyond this regime leads the solvers to be far less practical. Many variants of IPMs exist; the runtime of the best provably correct classical IPMs for the regime where the number of constraints is roughly equal to the number of degrees of freedom is

$$\begin{aligned} \text{LP [304]:} &\quad \widetilde{O}\left(n^{\omega}\log(1/\epsilon)\right) \\ \text{SOCP [778]:} &\quad \widetilde{O}\left(n^{\omega+0.5}\log(1/\epsilon)\right) \\ \text{SDP [538]:} &\quad \widetilde{O}\left(n^{2\omega}\log(1/\epsilon)\right), \end{aligned}$$

[3] In practice, these solvers are not solely based on IPMs and utilize many methods at once. Additionally, they employ heuristic preprocessing methods to transform and simplify inputs prior to applying an IPM. Note that they are useful also for nonconvex problems such as mixed-integer programs, where LP-solving can often be an important subroutine.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

where $\omega < 2.37$ is the matrix multiplication exponent. It is plausible that, with some attention, the extra $n^{0.5}$ factor for SOCP could be eliminated with modern techniques. Additionally, the runtime can be somewhat reduced when the number of constraints is much less than the number of degrees of freedom; for example, the n-dependence of the complexity of the CIPM for SDP in [574] can be as low as $\widetilde{O}(n^{2.5})$ when there are few constraints. On practical instances, employing techniques for fast matrix multiplication is often not beneficial, and Gaussian elimination–like methods are used, where n^ω is replaced with n^3. Note that, alternatively, by using iterative classical linear system solvers, such as the randomized Kaczmarz method [959], each n^ω factor could be replaced by a factor of n at the cost of a linear dependence on $(\kappa\zeta)^2$, which could be superior if the matrices are well conditioned.

For tall LPs ($m \gg n$), CIPMs can achieve scaling nearly linear in the number of matrix entries: the algorithm of [185] runs in time $O(mn+n^3)$.

We refer the reader to [1053, 1052, 797, 438] for additional information on CIPMs and their historical development.

(ii) Classical MWU methods: A classical complexity statement for LPs is inferred from the reduction in [46] from LPs to zero-sum games and the classical analysis that appears there. For the SDP case, references in the classical literature appear only to examine specific subclasses of SDPs (e.g., [57, 56]). A general statement of the classical complexity for SDPs appears alongside the quantum algorithm in [48, Section 2.4]:

$$\text{LP [46]:} \quad \widetilde{O}\left(s\left(\frac{rR}{\epsilon}\right)^{3.5}\right)$$

$$\text{SDP [48]:} \quad \widetilde{O}\left(snm\left(\frac{rR}{\epsilon}\right)^{4} + sn\left(\frac{rR}{\epsilon}\right)^{7}\right).$$

(iii) Cutting-plane methods: These classical methods are used for SDPs and can outperform IPMs when the number of constraints is small. The best algorithm, based on [671, 575], has runtime $O(m(mn^2 + n^\omega + m^2)\log(1/\epsilon))$, which can be as low as $O(n^\omega)$ when m is small.

It is important to note that the algorithms with the best provable complexities may not be the ones that are most useful in practice.

Speedup

For both the IPM approach and the MWU approach, there can be at most a polynomial quantum speedup: upper and lower bounds scaling polynomially

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

with n are known in both the classical and quantum cases [45]. The speedup of the QIPM method depends on the scaling of κ with n, but the speedup cannot be more than quadratic. For the MWU method, if $m = O(n)$, $s = O(1)$, and $rR/\epsilon = O(1)$, existing bounds on the classical and quantum complexities leave open the possibility of a quartic $\widetilde{O}(n^2) \to \widetilde{O}(\sqrt{n})$ speedup. However, it is unclear if the classical complexity quoted in [48, Section 2.4] is optimal (e.g., if the classical $\widetilde{O}(mn)$ scaling could be improved to $\widetilde{O}(m+n)$, the available quantum speedup would be at most quadratic). There is a possibility that the speedup could be larger in practice if the Gibbs sampling routine is faster on actual instances than its worst-case upper bounds suggest, perhaps by utilizing Monte Carlo–style approaches to Gibbs sampling.

Outlook

It is very plausible that an asymptotic polynomial speedup can be obtained in problem size using the MWU method for solving LPs or SDPs, but the speedup appears only quadratic, and an assessment of practicality depends on the scaling of certain unspecified instance-specific parameters. Similarly, the standard QIPM method could bring a subquadratic speedup but only under certain assumptions about the condition number of certain matrices. The alternative QIPM method of [51] could deliver a nearly quadratic speedup without assumptions on the condition number in the case the LP constraint matrix is very tall. In any case, these quadratic and subquadratic speedups alone might be regarded as unlikely to yield practical speedups after error correction overheads and slower quantum clock speeds are considered. Future work should aim to find additional asymptotic speedups while focusing on specific practically relevant use cases that allow the unspecified parameters to be evaluated.

5.3 General convex optimization

Overview

A convex problem asks to minimize a convex function f over a convex set K, where K is a subset of $\mathbb{R}^n$. Here we examine the situation where the value of $f(x)$ and the membership of x in the set K can each be efficiently computed classically. However, we do not exploit/assume any additional structure that may be present in f or K. This situation contrasts with that of solving conic programs, where f is linear and K is an intersection of convex cones and affine spaces, features that can be exploited to yield more efficient classical and quantum algorithms.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

A so-called "zeroth-order" solution to this problem solves it simply by adaptively evaluating $f(x)$ and $x \in K$ for different values of x. For the zeroth-order approach, a quantum algorithm can obtain a quadratic speedup with respect to the number of times these functions are evaluated, reducing it from $\widetilde{O}(n^2)$ to $\widetilde{O}(n)$, where $\widetilde{O}$ notation hides factors polylogarithmic in n and other parameters. This could lead to a practical speedup only if the cost to evaluate $f(x)$ and $x \in K$ is large, and lack of structure rules out other, possibly faster, approaches to solving the problem.

Actual end-to-end problem(s) solved

Suppose we have classical algorithms $\mathcal{A}_f$ for computing $f(x)$ and $\mathcal{A}_K$ for computing $x \in K$ ("membership oracle"), which require C_f and C_K gates to perform with a reversible classical circuit, respectively. Suppose further we have an initial point $x_0 \in K$ and that we have two numbers r and R for which we know that $B(x_0, r) \subset K \subset B(x_0, R)$, where $B(y, t) = \{z \in \mathbb{R}^n : \|z - y\| \leq t\}$ denotes the ball of radius t centered at y. Using $\mathcal{A}_f$, $\mathcal{A}_K$, x_0, r, R, and ϵ as input, the output is a point $\tilde{x} \in K$ that is ϵ-optimal, that is, it satisfies

$$f(\tilde{x}) \leq \min_{x \in K} f(x) + \epsilon \,.$$

Dominant resource cost/complexity

The work of [245] and [47] independently establish that there is a quantum algorithm that solves this problem with gate complexity upper bounded by

$$\left[(C_f + C_K)n + n^3\right] \cdot \text{polylog}(nR/r\epsilon) \,,$$

where the polylogarithmic factors were left unspecified. The rough idea behind the algorithm is to leverage the quantum gradient estimation algorithm to implement a *separation oracle*—a routine that determines membership $x \in K$ and when $x \notin K$ outputs a hyperplane separating x from all points in K—using only $O(1)$ queries to algorithm $\mathcal{A}_K$ and $\mathcal{A}_f$. It had been previously established that $\widetilde{O}(n)$ queries to a separation oracle then suffice to perform optimization [671], where $\widetilde{O}$ denotes that logarithmic factors have been suppressed.

Existing resource estimates

There have not been any explicit resource estimates for this algorithm. It may not make sense to perform such an estimate without a more concrete scenario in mind, as the estimate would highly depend on the complexity of performing the circuits for $\mathcal{A}_f$ and $\mathcal{A}_K$. The estimate would also require a more detailed accounting of the hidden polylogarithmic factors in the complexity statements above, and it would only be meaningful if the comparable classical complexity

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

for solving the same problem using the best available algorithm were well understood.

Caveats

One caveat is that the quantum algorithm must coherently perform reversible implementations of the classical functions that compute $f(x)$ and $x \in K$. Compared to a nonreversible classical implementation, this may cost additional ancilla qubits and gates. Another caveat relates to the scenario where $f(x)$ and $x \in K$ are determined by classical data stored in a classical database. Such a situation may appear to be an appealing place to look for applications of this algorithm because when f and K are determined empirically rather than analytically, it becomes easier to argue that there is no structure that can be exploited. However, in such a situation, implementing $\mathcal{A}_f$ and $\mathcal{A}_K$ would require a large gate complexity, so C_f and C_K would scale with the size of the classical database. It would almost certainly be the case that a quantum random access memory (QRAM) admitting log-depth queries would be needed in order for the algorithm to remain competitive with classical implementations that have access to classical RAM, and the practical feasibility of building a large-scale log-depth QRAM has many additional caveats.

Another caveat is that there may not be many practical situations that are compatible with a quantum speedup by this algorithm. The source of the speedup in [245, 47] comes from a separation between the complexity of computing the gradient of f classically vs. quantumly using calls to the function f. Classically, this requires at least linear-in-n number of calls. Quantumly, it can be done in $O(1)$ calls using the quantum algorithm for gradient estimation. In both the classical and the quantum case, the gradient can subsequently be used to construct a "separation" oracle for the set K, which is then used to solve the convex problem.

Thus, a speedup is only possible if there is no obvious way to classically compute the gradient of f other than to evaluate f at many points. This criterion is violated in many practical situations, which are often said to obey a "cheap gradient principle" [457, 163] that asserts that the gradient of f can be computed in time comparable to the time required to evaluate f. For example, the fact that gradients are cheap is crucial for training modern machine learning models with a large number of parameters. When this is the case, the algorithms from [245, 47] do not offer a speedup. On the other hand, as observed in [47, Footnote 19] a nontrivial example of a problem where the cheap gradient principle may fail (enabling a possible advantage for these quantum algorithms) is the moment polytope problem, which has connections to quantum information [211].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

When both the function f and the gradient of f can be evaluated at unit cost, this constitutes "first-order" optimization, which can be solved classically by gradient descent. However, gradient descent does not generally offer a quantum speedup, as general quantum lower bounds match classical upper bounds for first-order optimization, although a quantum speedup could exist in specific cases [410]. Indeed, for any p, there is no general quantum speedup for pth-order optimization, that is, the setting where an oracle provides access to the function and its first p derivatives [409]. For a comprehensive exposition of classical methods for black-box optimization, see [798].

Comparable classical complexity

The best classical algorithm [672] in the same setting has complexity

$$\left[(C'_f + C'_K)n^2 + n^3\right] \cdot \text{polylog}(nR/r\epsilon)\,,$$

where C'_f and C'_K denote the classical complexity of evaluating f and querying membership in K, respectively, without the restriction that the circuit be reversible.

Speedup

The speedup is greatest when quantities C_f and C_K are large compared to n and roughly equal to C'_f and C'_K. In this case, the quantum algorithm can provide an $O(n)$ speedup, which is at best a polynomial speedup. The maximal power of the polynomial would be obtained if $C_f + C_K \approx C'_f + C'_K$ scales as n^2, corresponding to a subquadratic speedup from $O(n^4)$ to $O(n^3)$.

Outlook

The only analyses of this strategy are theoretical in nature, interested more so in the query complexity of solving this problem than any specific applications it might have. As such, the analysis is not sufficiently fine-grained to determine any impact from constant prefactors or logarithmic prefactors. While a quadratic speedup in query complexity is possible, the maximal speedup in gate complexity is smaller than quadratic. Moreover, there is a lack of concrete problems that fit into the paradigm of "structureless" quantum convex optimization. Together, these factors make it unlikely that a practical quantum advantage can be found in this instance.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

5.4 Nonconvex optimization: Escaping saddle points and finding local minima

Overview

Finding the global minimum of nonconvex optimization problems is challenging because local algorithms get stuck in local minima. In analogy to physics where the objective function is the energy of the system, these local minima are stable configurations that locally optimize the energy but do not achieve the globally minimal energy. Often, there are many local minima and they are each separated by large energy barriers. Accordingly, instead of finding the global minimum, one may settle for finding a local minimum: local minima can often still be used effectively in situations such as training machine learning models. An effective approach to finding a local minimum is gradient descent, but gradient descent can run into the problem of getting stuck near saddle points, which are not local minima but nonetheless have a vanishing gradient. Efficiently finding local minima thus requires methods for escaping saddle points. Limited work in this area suggests a potential polynomial quantum speedup [1081] in the dimension dependence for finding local minima, using subroutines for Hamiltonian simulation and quantum gradient estimation.

Actual end-to-end problem(s) solved

Suppose we have a classical algorithm $\mathcal{A}_f$ for (approximately) computing a function $f : \mathbb{R}^n \to \mathbb{R}$ which requires C_f gates to perform with a reversible classical circuit. The amount of error tolerable is specified later. Following [1081], suppose further that f is ℓ-smooth and ρ-Hessian Lipschitz, that is,

$$\|\nabla f(x_1) - \nabla f(x_2)\| \leq \ell \|x_1 - x_2\| \qquad \forall x_1, x_2 \in \mathbb{R}^n$$
$$\|\nabla^2 f(x_1) - \nabla^2 f(x_2)\| \leq \rho \|x_1 - x_2\| \qquad \forall x_1, x_2 \in \mathbb{R}^n,$$

where ∇f denotes the gradient of f (a vector), $\nabla^2 f$ denotes the Hessian of f (a matrix), and the norm notation $\|\cdot\|$ denotes the standard Euclidean norm for vector arguments and the spectral norm for matrix arguments.

The end-to-end problem solved is to take as input a specification of the function f, an initial point x_0, and an error parameter ϵ, and to output an ϵ-approximate second-order stationary point (i.e., approximate local minimum) x, defined as satisfying

$$\|\nabla f(x)\| \leq \epsilon \qquad \lambda_{\min}(\nabla^2 f(x)) \geq -\sqrt{\rho\epsilon},$$

where $\lambda_{\min}(\cdot)$ denotes the minimum eigenvalue of its argument. In other words, the gradient should be nearly zero, and the Hessian should be close to a positive-semidefinite matrix.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost/complexity

The idea pursued in the quantum algorithm of [1081] is to run normal gradient descent, which has gradient query cost independent of n, until reaching an approximate saddle point. Classical algorithms typically apply random perturbations to detect a direction of negative curvature and continue the gradient descent. Instead, the quantum algorithm constructs a Gaussian wavepacket localized at the saddle point, and evolves according to the Schrödinger equation

$$\mathrm{i}\frac{\partial}{\partial t}\Phi = \left(-\frac{1}{2}\Delta + f(x)\right)\Phi\,, \tag{5.1}$$

where Δ denotes the Laplacian operator. The intuition is that, in the directions of positive curvature, the particle stays localized (as in a harmonic potential), while in the directions of negative curvature, the particle quickly disperses. Thus, when the position of the particle is measured, it is likely to have escaped the saddle point in a direction of negative curvature, and gradient descent can be continued. The other technical ingredient is the quantum gradient estimation algorithm, which uses a constant number of (coherent) queries to the function f to estimate ∇f.

The main finding of [1081] is that, to do the gradient descent and the Hamiltonian simulation, the algorithm need only query the circuit evaluating the function f polylog(n) times. Specifically, the algorithm performs the C_f-gate quantum circuit for coherently computing f a number of times scaling as

$$\widetilde{O}\left(\frac{\log(n)(f(x_0)-f^*)}{\epsilon^{1.75}}\right),$$

where x_0 is the initial point and f^* is the global minimum of f. The evaluation of f must be correct up to precision $O(\epsilon^2/n^4)$. Note that the work of [1081] initially showed a $\log^2(n)$ dependence, which was later improved to $\log(n)$ using the improved simulation method of [287, Corollary 8]. However, it is important to emphasize that the method has additional cost beyond the queries to the circuit for evaluating f, originating from the Hamiltonian simulation of the kinetic term $-\frac{1}{2}\Delta$ in Eq. (5.1). Specifically, the number of additional gates needed in the Hamiltonian simulation is seen from [287, Lemma 12 & Corollary 8] to scale as $\widetilde{O}(n(f(x_0)-f^*)/\epsilon^{1.75})$, although we remark that it is possible that this gate complexity could be parallelized such that the circuit depth scales polylogarithmically in n. Thus, the overall gate complexity is

$$\widetilde{O}\left(\frac{(n+C_f\log(n))(f(x_0)-f^*)}{\epsilon^{1.75}}\right).$$

The space complexity to represent the d-dimensional system on a grid with grid spacing $O(\epsilon)$ is $O(n\log(1/\epsilon))$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

In related work, [439] analyzes the complexity of escaping a saddle point when one has access to *noisy* queries to the value of the function f. Additionally, lower bounds on the ϵ-dependence of quantum algorithms for this problem are given in [1080].

Existing resource estimates

This problem has received relatively little attention, and no resource estimates have been performed.

Caveats

Reference [1081] gives the query complexity of the quantum algorithm but does not perform a full end-to-end resource analysis. (However, it does numerically study the performance of the quantum algorithm in a couple of toy examples.) Additionally, many practical scenarios are said to obey a "cheap gradient principle" [457, 163], which says that computing the gradient is almost as easy as computing the function itself, and in these scenarios, no significant quantum speedup is available. Finally, in the setting of variational quantum algorithms, this does not avoid the issue of barren plateaus, which refers to the situation where a large portion of the parameter space has a gradient (and Hessian) that vanishes exponentially with n. These regions would be characterized as ϵ-approximate local minima unless ϵ is made exponentially small in n.

Comparable classical complexity and challenging instance sizes

The best classical algorithm [1079] for this problem makes

$$\widetilde{O}\left(\frac{\log(n)(f(x_0) - f^*)}{\epsilon^{1.75}}\right)$$

queries to the *gradient* of f. Note that $\Omega(n)$ queries to the value of f would be needed to construct a query to the gradient. (When the quantum algorithm in [1081] was first discovered, the best classical algorithm required $O(\log(n)^6)$ gradient queries [577, Theorem 3], and this was later improved.) Although the literature focuses mainly on the query complexity, examination of [1079, Algorithm 1] indicates that an additional $\widetilde{O}(n(f(x_0)-f^*)/\epsilon^{1.75})$ arithmetic operations would be required to process the results of the gradient queries and compute the next point to be queried (e.g., adding pairs of n-dimensional vectors).

Speedup

The quantum algorithm in [1081] has the same query complexity as the classical algorithm in [1079]; the difference is that the quantum algorithm makes

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

(coherent) queries to an evaluation oracle, while the classical algorithm requires access to a gradient oracle. Thus, if classical gradient queries are just as cheap as evaluation queries (as is often the case), there is no speedup. If it were the case that gradient queries are not directly available, then the speedup in query complexity could be exponential. However, even in this case, the speedup in gate complexity can be at most polynomial, since both the classical and quantum algorithms have poly(n) gate complexity, and classical gradient queries can be constructed from $\widetilde{O}(n)$ classical evaluation queries, which can be accomplished with at most C_f classical gates. Indeed, the largest polynomial speedup in gate complexity occurs when $C_f = O(n)$—in this case, the quantum algorithm has $\widetilde{O}(n)$ gate complexity and the classical algorithm has $\widetilde{O}(n) \cdot C_f = O(n^2)$ gate complexity, a quadratic speedup.

Outlook

It is unclear whether the algorithm for finding local minima could lead to a practical speedup, as it depends highly on the (non)availability of an efficient classical procedure for implementing gradient oracles; a quantum speedup is possible only when such oracles are difficult to implement classically, and even so, the speedup in gate complexity would be modest. However, the algorithm represents a useful end-to-end problem where the quantum gradient estimation primitive can be applied. It is also notable that the quantum algorithm employs Hamiltonian simulation, a primitive not used in most other approaches to continuous optimization. Relatedly, [675, 674] propose a quantum subroutine called "quantum Hamiltonian descent" which is a genuinely quantum counterpart to classical gradient descent, via Hamiltonian simulation of an equation similar to Eq. (5.1). Unlike classical gradient descent, it can exploit quantum tunneling to avoid getting stuck in local minima; thus, it can potentially find *global* minima of nonconvex functions. Establishing concrete end-to-end problems where quantum approaches based on Hamiltonian simulation yield an advantage in nonconvex optimization is an interesting direction for future work.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

6

Cryptanalysis

Computation and communication are secured by cryptography. For example, a user's data can be made private, along with messages that they send or receive, from malicious agents who interfere to try to learn the sensitive information. A set of algorithms collectively called a *cryptosystem* endows the security. The attempt to break security is known as *cryptanalysis*, which has its own set of algorithms. Historically, both cryptography and cryptanalysis considered classical, polynomial-time algorithms as the only realistic ones. The advent of quantum computation forces us to consider attacks via quantum algorithms. Generally, we want to know what is the best algorithm for cryptanalysis, in order to understand the effect on the cryptosystem in the worst case. Quantum attacks can void the security of widely used cryptosystems (see Section 6.1 on breaking cryptosystems). More broadly, quantum cryptanalysis can reduce a cryptosystem's security (see Section 6.2 on weakening cryptosystems), such that it becomes more expensive to implement in a secure manner. While the properties of quantum mechanics can also be used to devise more secure cryptosystems (e.g., quantum key distribution) [121, 834, 1058], we consider this area of cryptography to be outside the scope of the present discussion on quantum algorithms.

The authors are grateful to Matthew Campagna and Samuel Jaques for reviewing this chapter.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

6.1 Breaking cryptosystems

Overview

Much of modern cryptography relies on computational assumptions.[1] A cryptosystem is a protocol to achieve some security goal, such as hiding information, ensuring integrity of information, or computing a function correctly. A cryptosystem is secure if, assuming a particular mathematical problem is hard to solve (or assuming the existence of certain functions, e.g., pseudorandom), an adversary cannot compromise the security goal. The earliest such cryptosystems used particular problems from number theory, and variants are widely deployed to this day [605]. These cryptosystems are in the class of public-key or asymmetric cryptography. Public-key cryptography uses key pairs: a private key known only to one user, and a public key that can be widely distributed to allow any user to perform tasks like encryption. In contrast, symmetric-key cryptography uses a single secret key that must be preshared between communicating parties.

Quantum computers use quantum algorithms to solve computational problems, and in some cases they provide a speedup over the best known classical techniques. When they are applied to the underlying computational task in a cryptosystem, a large speedup over classical methods can break the cryptosystem, in that an adversary efficiently learns the secret key or the encrypted information to a non-negligible degree. One of the first discovered and most famous applications of quantum computing is Shor's algorithm [937], which breaks or solves the integer factorization problem, and both the general discrete logarithm problem and the elliptic curve discrete logarithm problem. These problems are believed to be classically hard, and are the basis of security for the most common public-key cryptosystems, like Diffie–Hellman, Rivest–Shamir–Adleman (RSA), and elliptic curve cryptography (ECC). The applications of these public-key cryptosystems include encryption to hide the contents of a message, signatures that prevent tampering and impersonation, and key exchange to generate a key for symmetric-key cryptography [126]. In this section, we restrict our focus to two of the most widely used cryptosystems: RSA and ECC.

Actual end-to-end problem(s) solved

The RSA cryptosystem [875] relies on a user choosing a large number N that is the product of two prime numbers; arithmetic is done modulo N. Denote by $n = \lceil \log_2(N) \rceil$ the number of bits specifying N. Along with the modulus N, two integers e and d are used as exponents, such that $(m^e)^d = m \bmod N$ for all val-

[1] An example of a cryptosystem not requiring computational assumptions is the one-time pad.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ues $0 \leq m < N$. The pair (N, e) is the public key, and the value d is the private key. A message m is encrypted as $m^e \bmod N$. Exponentiation with d performs decryption, recovering m as specified in the relation above. Some applications store the factors of N, which must also be kept private. The user generates e and d using their knowledge of the prime factors of N. First, a suitable e is generated, and then d is computed from the prime factors of N and e using some number-theoretic facts together with the extended Euclidean algorithm. However, if an adversary is able to find the factors of N after the construction by the user, they can also solve for d and thereby decrypt messages. The security of RSA is based on the observed difficulty of factoring large numbers like N, that is, the integer factorization problem.

ECC is based on a different problem, the elliptic curve discrete logarithm problem (ECDLP), which has the advantage of smaller key sizes for equivalent security levels, compared to RSA. Consequently, fewer resources (e.g., communication, complexity of encryption and decryption) are required to implement ECC. Elliptic curves are constructed over a finite field K, as the set of solutions to the equation

$$y^2 = x^3 + ax + b\,, \quad a, b \in K\,,$$

which specify points on an elliptic curve [639, 763]. The set of solutions $P = (x, y)$ forms an abelian group under a specially defined addition operation. Collectively, the set of parameters K, a, b and a so-called base point G (a solution to the equation of prime additive order N, i.e., NG is the additive identity) specify the cryptosystem. A private key is a random integer k satisfying $1 \leq k < N$, and a public key is the value $P = kG$, the result of adding G to itself k times. The assumption of hardness is in the following problem (ECDLP): *Given P and G, where $P = kG$ for some secret value k, find k.* ECC is constructed from the observation that calculating P from k, G is efficient, whereas it is computationally infeasible for an adversary to compute k from points P and G.

Dominant resource cost/complexity

Shor's algorithm [937] solves the number-theoretic problem of order finding: given n-bit positive integer N and x coprime to N, find the smallest integer r such that $x^r = 1 \bmod N$. Factoring was shown to reduce to order finding. In particular, there is an efficient, otherwise classical algorithm, of classical complexity $O(n^3)$ [801], that uses order finding as a quantum subroutine. To describe the quantum algorithm for order finding, let the function f denote modular exponentiation, that is, $f(e) = x^e \bmod N$, and note that f is periodic with (unknown) period r. Also, let L be a large integer such that an interval of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

length L contains many periods, that is, $L \gg r$. It can be shown that $L \geq N^2$ is sufficient. There are three steps. First, an equal superposition over the numbers $\{0, \ldots, L-1\}$ is formed and the function f is computed into an ancilla register yielding the state $L^{-1/2} \sum_{e=0}^{L-1} |e\rangle|f(e)\rangle$. Second, a measurement is performed on the ancilla register, which, due to the periodicity of the function f, yields a state $(\lceil L/r \rceil)^{-1/2} \sum_{j=0}^{\lfloor L/r \rfloor} |rj + y\rangle$ for $0 \leq y < r$ a random and unknown integer.[2] Third, a quantum Fourier transform is performed. In the case that L is a multiple of r, the result is

$$\frac{\sqrt{r}}{L} \sum_{j=0}^{L/r} \sum_{z=0}^{L-1} \mathrm{e}^{2\pi \mathrm{i} z(rj+y)/L} |z\rangle = \frac{1}{\sqrt{r}} \sum_{\ell=0}^{r-1} \mathrm{e}^{2\pi \mathrm{i} \ell y/r} |\ell L/r\rangle \,, \tag{6.1}$$

where the equality follows since coefficients of $|z\rangle$ for which z is not equal to $\ell L/r$ for some integer ℓ vanish due to destructive interference. Measurement of this state then produces an outcome $\ell L/r$ for a randomly sampled ℓ. The value of r can be classically computed by dividing the measurement outcome by L and determining the value of the denominator of the rational number that results; repetition may be required since ℓ and r could have common divisors. If L/r is not an integer, the measurement outcome is (with high probability) an integer close to $\ell L/r$ for some integer ℓ. One can deduce the rational number ℓ/r (which allows for the determination of r) from the estimate of $\ell L/r$ by writing it as a continued fractions expansion, with classical complexity $O(n^3)$ [801].

This entire procedure can alternatively be viewed as quantum phase estimation applied to the unitary U that sends $|y\rangle \mapsto |xy \bmod N\rangle$ for all y relatively prime to N, performed with at least $2n$ bits of precision.

The number of qubits for order finding—and hence for Shor's factoring algorithm—is $O(n)$, which stems from the number of bits specifying the problem: the first register has size $2n$, and the ancilla register holding the result $f(e)$ has size n. Naively, the number of gate operations is $O(n^2)$ for the quantum Fourier transform and $O(n^3)$ for implementing the coherent modular exponentiation $|e\rangle|0\rangle \mapsto |e\rangle|x^e \bmod N\rangle$. The $O(n^3)$ arises from decomposing modular exponentiation into $O(n)$ modular multiplications, one for each bit of e—using schoolbook multiplication, the gate cost per multiplication is $O(n^2)$. Implementing this modular arithmetic with reversible circuits represents the bottleneck in the complexity. These circuits are closely related to those in classical computing that have been optimized. Still, improvements can lead to better resource counts [592]. The best scaling in theory is achieved with algorithms that have large prefactors in their complexity, making them impractical to implement except when n is large: total gate complexity of $O(n^2 \log(n))$ is possi-

[2] If $r\lfloor L/r \rfloor + y \geq L$, then the $j = \lfloor L/r \rfloor$ term does not appear in the expression.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ble asymptotically, using integer multiplication with $O(n\log(n))$ scaling [502], although analyses optimized for $n \approx 2048$ use methods for which the total gate complexity scales as $\widetilde{O}(n^3)$ [424]. Alternatively, optimization may be performed to, for example, increase qubit count and decrease gate count or gate depth. For example, an approximate version of the quantum Fourier transform is implemented with $O(n\log(n))$ gates and allows factoring with $O(\log(n))$-depth quantum circuits [298], at the cost of extra overhead in number of qubits and gates; allowing for $O(\log^2(n))$-depth preserves the circuit size $O(n^3)$.

A related approach proposed by Regev [869] for quantum factoring has quantum circuit size of only $\widetilde{O}(n^{3/2})$ gates (assuming fast integer multiplication at cost $\widetilde{O}(n)$), but the circuit has to be run $O(n^{1/2})$ times. Thus, it achieves the same overall asymptotic gate complexity as Shor's algorithm. The idea of the algorithm is to extend period finding to higher dimensions and optimize resource counts. Unlike Shor's algorithm, which is proven to succeed, Regev's approach relies on a number-theoretic assumption, albeit a plausible one. The reduction in quantum circuit depth and natural parallelism of the approach may lead to more favorable resource counts in practice. In particular, a constant fraction of the runs can fail [849], which may allow it to be implemented fault tolerantly with less overhead. Initial work on optimizing the algorithm has established a tradeoff between the number of qubits and the number of gates. Linear qubit cost of $O(n)$ is possible (although the constant prefactor is larger than that of Shor's algorithm) while still maintaining $\widetilde{O}(n^2)$ total gate complexity [849, 848].

Essentially the same quantum algorithm of Shor is readily applied to elliptic curves, as well as the discrete logarithm problem (i.e., find r such that $a^r = b$ for $a, b \in G$ where G is a group), which is also used as a computationally hard problem for cryptography. These applications are all instances of the *hidden subgroup problem*: Find the generators for subgroup K of a finite group G, given a quantum oracle performing $U|g\rangle|h\rangle = |g\rangle|h \oplus f(g)\rangle$, where $f : G \to X$ (X is a finite set) is a function that is promised to be constant on the cosets of K and take unique values on each coset. In the case of period finding, G is the group $\mathbb{Z}/L\mathbb{Z}$ under addition, and the hidden subgroup is $K = \{0, r, 2r, \ldots, L - r\}$ (technically a subgroup only if r divides L); one can verify that $f(g) = x^g \bmod N$ is constant on each coset of K. The procedure outlined above for period finding can be applied to other groups, where it is called "the standard method" [277], which requires generalizing the quantum Fourier transform to arbitrary groups. A simple example is Simon's problem [940]—indeed, historically speaking Simon's algorithm inspired Shor's [938]—where G is the abelian group $(\mathbb{Z}/2\mathbb{Z})^n$ of bit strings of length n under addition, $K = \{0, c\}$ for some hidden bit string c, and the generalized Fourier

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

transform is simply the Hadamard transform $H^{\otimes n}$. For abelian groups, the hidden subgroup K can be determined with polylog($|G|$) queries to f, but the method does not work for nonabelian groups, such as the symmetric group and the dihedral group.

Existing resource estimates

The minimum recommended key size for RSA is 2048 bits [96]. Optimizations in the circuits [109, 493] and incorporation of hardware constraints [398] have led to decreasing but also more realistic resource estimates. For key size $n = 2048$, an optimized resource estimate was performed in [424], geared toward implementation on a device with nearest-neighbor connectivity in 2D, where logical qubits are encoded with the surface code. Their implementation used roughly $3n \approx 6000$ logical qubits and roughly $0.3n^3 \approx 3 \times 10^9$ non-Clifford gates (a mixture of Toffoli and T gates). Accounting for magic state distillation and routing in 2D, it was shown how the computation could be laid out on a 2D grid of 14,000 logical qubits.

For ECC, the minimum recommended key size to ensure 128-bit security (quantifying the number of attacks needed to learn the encrypted information; see Section 6.2 on weakening cryptosystems for details), is $n = 256$ bits [96] (achieving the same level of security with RSA requires a key size of 3072 bits [175, 876]). For breaking 256-bit ECC, an early resource esimate concluded that around three times fewer logical qubits, and 100 times fewer Toffoli gates are required (compared to 3072-bit RSA) [876]. Similar to factoring, improvements have been made in logical circuit compilation [495] and how this translates into hardware implementations [1030, 450, 695]. In [695], a method was given requiring 1.1×10^8 Toffoli gates and 6000 logical qubits. Additionally, by offloading some of the work to a brute-force classical computation and exploiting a simplification that arises when computing multiple ECC keys in parallel, the total Toffoli count per key was shown to approach 4.4×10^7 [695]. As a conclusion, breaking elliptic curve cryptography is easier than factoring for quantum computers in practice [846], relative to their practical difficulty on classical computers.

The physical resources required to implement these logical circuits fault tolerantly depends on many details of the hardware, including the error rate, the physical gate speed, and the available connectivity. In both cases (2048-bit RSA [424, 476] and 256-bit ECC [1030, 450, 695]), given current hardware schemes restricted to nearest-neighbor 2D connectivity with logical qubits encoded into surface codes, the number of physical qubits is estimated to be on the order of 10 million and the computation runs for at least 3–10 hours (significantly longer than this for platforms with relatively slower physical gate

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

speeds). For a discussion on how to convert between logical and physical resources, see Part III on fault-tolerant quantum computation. Optimization based on the particular architecture can give improvements to these estimates. For example, if one assumes a logarithmic number of nonlocal links, as could be envisaged in photonic implementations, the estimated runtime can be reduced to less than one minute per 256-bit ECC key [695]. The algorithms considered in the resource estimates above do not achieve the best known asymptotic scaling, which comes at the cost of large constant prefactors.

Caveats

While the popular cryptosystems based on number-theoretic problems are rendered insecure for public-key cryptography, there exist alternatives that are believed to be secure against quantum computers: for example, based on error correcting codes or lattices [126]. These alternative computational problems are believed to be hard for both classical and quantum computers. The National Institute of Standards and Technology (NIST) of the United States has provided standards and encouraged implementation [15]. The class of symmetric-key cryptography (see a standard text [605] for details) involves computations that do not have much structure, and also is not broken by quantum computers.[3] Instead, the number of bits of security is reduced.

Prior experimental demonstrations of Shor's algorithm have used knowledge of the answer in order to optimize the circuit and thus lead to sizes that are experimentally feasible on non-error-corrected devices. Meaningful demonstration should avoid such shortcuts [943], which are not available in realistic cryptographic scenarios.

Comparable classical complexity and challenging instance sizes

The best known classical algorithm for factoring is the number field sieve, which has time complexity superpolynomial in number of bits n: namely, it scales as $O(\exp(p \cdot n^{1/3} \log^{2/3}(n)))$, where $p > 1.9$. With a hybrid quantum-classical algorithm applying amplitude amplification on the number field sieve, $p = 1.387$ can be achieved using a number of qubits scaling only as $O(n^{2/3})$ [128]. Classically, problems of size 795 bits have been factored, taking 76 computer core-years, which distributed in parallel over a cluster took 12 days; the same team then extended the record to 829 bits [175].

Several algorithms attacking elliptic curve cryptography have complexity $O(2^{n/2})$ [1026], leading to the recommended doubling of key size compared to

[3] If the adversary can query the cryptosystem's algorithms in superposition, some symmetric-key cryptography can be broken using period finding to extract the key [599]. However, this capability of the adversary is not considered realistic.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

bits of security. In practice, a problem of size 117 bits was solved [127]. The corresponding security is estimated [676] to be about 60 bits, compared to 70 bits for the RSA record above.

Speedup

The number of gates to implement Shor's algorithm is $\widetilde{O}(n^2)$ asymptotically using fast multiplication on large numbers [111]. More practically, without incurring the time overhead and additional storage space of fast multiplication, the scaling is $O(n^3)$. Assuming classical and quantum gates are polynomially related in time complexity, the speedup for solving the factoring problem is superpolynomial, and the speedup for solving the ECDLP is exponential. However, there are no tight lower bounds on the classical complexity of factoring or ECDLP; it remains possible that more efficient classical algorithms could be discovered.

NISQ implementations

The large circuit depth, complicated operations, and high number of qubits needed to implement Shor's algorithm make faithful NISQ implementation challenging. However, there have been several attempts to ease implementation at the expense of losing the guarantees of Shor's algorithm, in the hope that the output is still correct with some nonzero probability, which could be vanishing.

One approach [885] is to simplify several operations and make them approximate. The outcome is that the circuit depth is $O(n^2)$, saving a factor of n [493]. The depth is then about 10^8 to factor a 1024-bit instance of RSA, so for relevant sizes, error correction is still required. Implementation of the approximate algorithm, including experimentally, allowed for the successful factorization of larger problem instances than had been possible before. This approximate version is not NISQ in the usual sense of involving noisy circuits, but rather introduces some uncontrolled approximation error in return for reducing the depth, for the possibility of a useful result. Another approach is to encode the factoring problem in a variational optimization circuit. Again, performance is not guaranteed; moreover, variational optimization applied to generic problems is expected to have, at best, a quadratic improvement compared to classical methods, leaving no hope for breaking cryptography. Classical simulation on small problem sizes shows that the algorithm can succeed [39], as does experimental implementation on a superconducting quantum processor [600]. We emphasize that, generally, these NISQ approaches have no evidence or arguments for scaling to cryptographically relevant system sizes.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Outlook

The existence of Shor's algorithm implies common RSA and elliptic curve schemes are theoretically not secure, and resource estimates have made clear what scale of quantum hardware would break them. While such hardware does not exist currently, progress toward such a device can be used to inform the speed of transitioning to quantum-resistant encryption [263]. Currently, from a hardware perspective, the field of quantum computing is far from implementing algorithms that would break encryption schemes used in practice. The estimates above suggest that the resources required would be millions of physical qubits performing billions of Toffoli gates running on the timescale of hours or days. In contrast, the current state of the art is on the order of one hundred noisy physical qubits, with progress toward demonstration of a single logical qubit. Running fault-tolerant quantum computation requires extra overhead, such as magic state factories (see Chapter 26 on quantum error correction and Chapter 27 on lattice surgery). Thus, the gap between state-of-the-art hardware and the requirements for breaking cryptosystems is formidable. Moreover, a linear increase in key size will increase, for example, the number of Toffoli gates by a power of three, which can be substantial. Therefore, considering the experimental challenges, likely only the most sensitive data will be at risk first, rather than common transactions. Consequently, these highly confidential communications will likely adopt post-quantum cryptography first to avoid being broken. However, insecure protocols often linger in practice, so quantum computers can exploit any vulnerabilities in deployed systems that have not been addressed. For example, RSA keys of size 768 bits have been found in commercial devices (note that such key sizes can already be broken classically [175]). In addition, intercepted messages, encrypted with RSA or elliptic curves, can be stored now and decrypted later, once large-scale quantum computers become available.

The resilience of candidates for post-quantum cryptography is under active investigation. In particular, specialized quantum attacks [274, 831] can reduce the number of bits of security, weakening the cryptosystem. Relaxing to toy variants of relevant cryptosystems in order to find new attacks, quantum algorithms can provide polynomial-time solutions [318]. Classical algorithms have even broken certain candidate cryptosystems [145, 239]. Note that these attacks affect the feasibility of particular proposals, but there exist other post-quantum candidates that have no known weaknesses. With the completion of NIST's standardization process, approved post-quantum cryptography is being rapidly deployed. For example, several popular messaging platforms [645, 378] recently adopted post-quantum key derivation hybridized

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

with ECC, so that the scheme is secure as long as at least one of the underlying cryptosystems is secure.

A sensitive area that warrants additional discussion is cryptocurrency, since much of it relies on the compromised public-key cryptography based on abelian groups. Moreover, changing the cryptographic protocol of the currency requires that most of the users reach a consensus to do so, which can be challenging to coordinate, even if the technical hurdles of adopting post-quantum encryption are overcome. Cryptocurrency wallets that have revealed their public key (e.g., via a transaction reusing a public key assigned to that wallet previously) can be broken using Shor's algorithm. An attack is also possible during the short time window in which the key is revealed during a single transaction [8]. Different cryptocurrencies have different levels of susceptibility to these types of attacks [97, 98]. Nevertheless, the mining of cryptocurrency is not broken, but only weakened by quantum computers.

6.2 Weakening cryptosystems

Overview

The discovery of Shor's algorithm (see Section 6.1 on breaking cryptosystems) prompted interest in post-quantum cryptography, the study of cryptosystems assuming the presence of large-scale, working quantum computers [126]. While some existing systems retained confidence in their security, others that were broken by quantum algorithms were superseded by those that accomplish the same task, but are believed to maintain a high level of security against quantum attacks.

Even if a cryptosystem is not broken altogether, its degree of security can be weakened by quantum algorithms. The strength of a cryptosystem is typically quantified by the number of bits of security (also called the security parameter), that is, n bits corresponds to guessing the desired information with probability $1/2^n$ and accessing what is being protected. When considering computational assumptions, a simplified definition of the security parameter n is that cryptanalysis requires a computational cost of 2^n operations, captured by the best known attack. *Breaking* a cryptosystem means only an efficient number of operations (i.e., $\text{poly}(n)$) are needed, while an attack that *weakens* a cryptosystem still takes $2^m > \text{poly}(n)$ operations, for some $m < n$.

In contrast to public-key cryptosystems, symmetric-key cryptography was discovered earlier and has fewer capabilities. However, it relies less on the presumed hardness of underlying mathematical problems, and correspondingly

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

has only been weakened by quantum cryptanalysis, as discussed in more detail below.

Actual end-to-end problem(s) solved

In symmetric-key cryptography, two communicating parties share the same key K, which is used both in encryption Enc_K and decryption Dec_K. As usual, the cryptographic algorithm (Enc_K, Dec_K) is known to everyone, including adversaries. The most significant break of a symmetric-key algorithm is an adversary learning the key, given r pairs of plaintext (the message m) and corresponding ciphertext c (its encryption).[4] Such a pair can be accessed by, for example, forcing a certain test message to be transmitted. Precisely, an input K is sought for which the following function outputs 1:

$$f(K) = ((Enc_K(m_1) = c_1) \wedge \cdots \wedge (Enc_K(m_r) = c_r)),$$

that is, find a key such that all the messages encrypt correctly. A straightforward attack is to use brute force and test every key; in practice, sophisticated classical attacks do not perform better than this approach in asymptotic scaling.

Dominant resource cost/complexity

The main, generic quantum attack is to use amplitude amplification: given a classical algorithm with success probability $O(2^{-n})$ of finding a solution, the probability is increased quadratically to $O(2^{-n/2})$. Thus, applying amplitude amplification to the task of solving for the key, the security of cryptosystems goes from n bits to $n/2$.

The function queried in superposition must be efficient to evaluate with a quantum circuit, which is often the case in cryptography [126]. However, the operations are typically long sequences of Boolean arithmetic. As such, a universal gate set and fault-tolerant quantum computation are still required. To store the key, $O(n)$ register qubits are needed, and many more ancilla qubits are used for the reversible arithmetic.

Existing resource estimates

Consider the Advanced Encryption Standard (AES) [559], a symmetric encryption algorithm that is widely used in cryptosystems, for example, for encrypting web traffic. At a high level, it mixes the plaintext and adds it to the key to obtain the ciphertext. An attack based on amplitude amplification needs

[4] There are more sophisticated attack models: for example, many communicating adversaries may have the goal of compromising one of multiple keys [89]. Correspondingly, there are other definitions of security, but this simple and powerful one generally is considered for quantum attacks.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

around 3000–7000 logical qubits [453] for AES-k, where k denotes key size in bits, and $k \in \{128, 192, 256\}$. For these sizes, the number of necessary problem instances r is three to five. While the number of logical qubits roughly doubles going from AES-128 to AES-256, the number of T gates goes from $2^{86} \approx 10^{25}$ to $2^{151} \approx 10^{45}$. More recent work [566] allows for higher qubit counts (by about 70%) in exchange for much lower Toffoli depth, such that their product is reduced. Such optimization resulted in about 99% reduction in this metric.

Caveats

Since the quantum attack only halves the exponent in the complexity, a simple fix is to double the key length, for example, by adopting AES-256 instead of AES-128. This modification results in increased, but usually tolerable, cost in implementation (i.e., complexity of encryption and communication resources). In addition, there exist cryptosystems with an information-theoretic security guarantee, assuming adversaries with unlimited computational power, which covers against quantum attacks [126].

Furthermore, it is important to note that to realize the full quadratic benefit of amplitude amplification, the $O(2^{n/2})$ function queries must be performed in series. In contrast, classical brute-force attacks can exploit the parallelism available in high-performance classical computers, potentially increasing the value of n for which a quantum approach would be advantageous over classical methods.

Comparable classical complexity and challenging instance sizes

Classical algorithmic attacks on AES have reduced the security by only a few bits [160]. More practical are side-channel attacks, which make use of physical byproducts, such as energy consumption. For example, when comparing bits between a key and another string, a flipped value can result in logic that increases energy consumption, compared to the same value where nothing happens. The two cases are distinguished and information about the key is learned. Currently, 128 bits of security is roughly the minimum recommended amount [95]. The use of parallelization forces the adoption of relatively large key sizes, compared to what is necessary for a single processor ($\sim$ 60 bits).

Speedup

The basic speedup is quadratic: $O(\sqrt{N})$ function evaluations compared to $O(N)$ classically, where N denotes the number of possibilities for the key; that is, $n = \lceil \log_2(N) \rceil$. However, the function queries in amplitude amplification cannot be parallelized. Then, the evaluation time of the function sets a bottleneck [126]. That is, the problem size is limited by the number of function

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

evaluations T that can be run in an acceptable period of time. For $\sqrt{N} > T$, employing p parallel quantum processors, each executes $T = \sqrt{N/p}$ evaluations. Then, $p = O(N/T^2)$ and the total number of evaluations is $pT = O(N/T)$, whereas classically, the number of processors is $O(N/T)$ and total evaluations is $O(N)$. The advantage is a factor of T, which is the bottleneck, rather than the larger $\sqrt{N}$. However, the advantage can be overshadowed by faster or cheaper classical processing. That is, if classical computers evaluate the function T times faster than quantum processors, there is no advantage in runtime with using the quantum device. Furthermore, this argument assumes the same cost of parallelization for classical and quantum, which is optimistic for quantum devices. An example of this effect is in mining cryptocurrency [8]: while a quantum computer needs quadratically fewer attempts to succeed, the development of fast, specialized, classical hardware negates the advantage. Essentially, for brute-force attacks, parallelization has the most significant impact in cryptanalysis.

NISQ implementations

The key can be encoded as the ground state of a Hamiltonian, and then variational methods can be applied to solve for it. The scaling is expected to be the same as amplitude amplification. However, since the variational algorithm does not have a set time complexity, the solution may be found much slower or faster [1023]. If the fluctuations are large enough, they can potentially pose a challenge to cryptography, which makes worst-case guarantees. However, there is no reason to expect that the success probability will scale favorably with key size and compromise security in practice. Another approach is to use amplitude amplification, but adapt it to near-term devices, so that the NISQ-optimized versions perform better in real experiments [1083].

Outlook

Here, we focused on the example of symmetric-key encryption. Nonetheless, the effect of amplitude amplification to halve the effective bits of security is generic for computational problems, assuming efficient construction of the oracle. From the cryptographic standpoint, this attack is mild and can be counteracted by doubling the number of bits of security in the scheme. In practice, the increase in key size can be unwieldy in certain applications, such as cryptocurrencies, but fundamental security is not threatened.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

7

Solving differential equations

The authors are grateful to Dong An, Di Fang, and Ashley Montanaro for reviewing this chapter.

Overview

Many applications in engineering and science rely on solving differential equations. Accordingly, this constitutes a large fraction of research-and-development high-performance computing (HPC) workloads across a wide variety of industries. There have been many proposals to speed up differential equation solving using a quantum computer. At this point, the consensus is that we lack compelling evidence for practical quantum speedup on industry-relevant problems. However, the field is progressing rapidly, and this conclusion is subject to change.

Some of the main application areas that have been considered are:

- **Computational fluid dynamics** (CFD), usually involving simulation of the Navier–Stokes equation. The main industries relying on CFD simulations are automotive, aerospace, civil engineering, wind energy, weather/climate modeling, and defense. While most simulations focus on air or fluid flow on solid objects, other processes, such as foaming, are also important to model. Large CFD calculations are routinely in the petaflop regime and are run on millions of CPU cores. Specific quantum proposals include [683, 965, 560, 584, 821, 406, 405, 268, 661].
- **Geophysical modeling**, involving simulation of the wave equation. The main industries are oil and gas, hydroelectric, geophysics. Large seismic imaging simulations can easily be in the petaflop regime. Quantum proposals for simulating the wave equation include [779, 518, 365].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- A wide variety of engineering problems involving the **finite element method** (FEM) for studying structural properties of solid objects. The main industries are civil engineering, manufacturing (including automotive), aerospace, and defense. The simulations are typically slightly smaller in scale than CFD, though still often requiring large HPC clusters. Quantum FEM proposals include [580, 777, 1002, 1082].
- **Maxwell's equations and the heat equation** have applications in chip design and other electronic component design, as well as for navigation and radar technology. Specific quantum proposals include [295, 580, 692].
- **Plasma physics** simulations, involving the simulation of the Vlasov equation, are widespread in nuclear fusion research. Quantum approaches include [803, 377, 354].
- **Risk modeling**, involving the simulation of stochastic differential equations, is extensively used in finance (especially derivatives pricing), insurance, and energy markets. Specific quantum proposals include [865, 33, 857, 393, 685].

Differential equations can be categorized according to a number of properties: (i) *ordinary vs. partial* depending on the number of differential variables, (ii) *stochastic vs. deterministic* depending on whether the function is a random variable or not, (iii) *linear vs. nonlinear*. We will focus mainly on linear ordinary and partial differential equations, which have received the most attention in the quantum computing literature, and only comment briefly on stochastic and nonlinear differential equations.

In order to solve a differential equation numerically, one typically specifies a discretization scheme. Two important classes of discretization schemes are: (i) *grid-based* schemes, including finite difference methods (FDMs), as well as the finite volume method (FVM) and the FEM combined with various choices of support grids and preconditioning (see [665, 996] for an introduction). For example, in the finite difference framework, the continuous space is discretized on a uniform grid and the continuous operators are replaced by finite difference operations on neighboring grid points. Alternatively, (ii) one can discretize space by expansion in a functional basis (Fourier, Hermite, etc.), and solve the discrete problem in this basis. This second class of methods is often referred to as *spectral methods* [930]. There is often a tradeoff between error convergence and regularity requirements, with higher-order grid-based methods and spectral methods offering faster error convergence with the number of grid points or basis functions utilized, but requiring more stringent assumptions on the smoothness of the solution of the differential equation, which are not always satisfied in the applications listed above.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Given a discretization scheme, solving linear differential equations can be accomplished by solving linear systems of equations. In cases where one is interested in very high precision, requiring very fine discretization, the linear system of equations can be too large for straightforward numerical solutions on a classical computer. In particular, if one wants high-precision results integrated over time, and/or systems with many continuous variables, then the simulations can be challenging both in time and memory. Quantum algorithms aim to offer a speedup over classical methods by leveraging the existence of quantum linear system solvers, or more generally, primitives in quantum linear algebra, which enable quantum algorithms to perform efficient manipulations of vectors that are exponentially large in the number of qubits and elementary operations involved. However, at a technical level, various complications arise, including the difficulty of reading out useful information at the end of the algorithm, and assumptions about the differential equation that must be true for the methods to work. Ultimately, polynomial speedups for end-to-end problems appear to be possible, but for differential equations in a fixed number of spatial dimensions, exponential speedups for real-world applications are not generally attainable.

Actual end-to-end problem(s) solved

We are interested in solving a general linear partial differential equation (PDE) of the form

$$\mathcal{L}(u(x)) = f(x) \quad \text{for} \quad x \in \mathbb{C}^d\,, \tag{7.1}$$

where $\mathcal{L}$ is a linear differential operator acting on the function $u(x)$, and $f(x) \in \mathbb{C}$ is the inhomogeneous term (which is 0 for homogeneous PDEs). In addition to Eq. (7.1), we are given boundary conditions on $u(x)$ and its derivatives, which ideally are sufficient to ensure a unique solution—for example, Dirichlet boundary conditions refer to a specification of a function $b(x)$ and a requirement that $u(x) = b(x)$ for x contained in some subset $\Omega \in \mathbb{C}^d$. As a canonical example of a linear PDE, consider the Poisson equation in d dimensions, given by

$$\frac{\partial^2 u}{\partial x_1^2} + \cdots + \frac{\partial^2 u}{\partial x_d^2} = f(x)\,. \tag{7.2}$$

As another example, we consider a linear ordinary differential equation (ODE) of the form

$$\frac{\mathrm{d}\bar{u}(t)}{\mathrm{d}t} = A(t)\bar{u}(t) + \bar{b}(t)\,, \tag{7.3}$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

where we refer to the variable $t \in \mathbb{R}$ as time (although it could represent a different quantity), $\bar{u}(t)$ and $\bar{b}(t)$ are N-dimensional vectors, and $A(t)$ is an $N \times N$ matrix. Boundary conditions on $\bar{u}(t)$ are also specified, often in the form of an initial condition at $t = 0$, with one seeking the solution at some final time T; this is known as an *initial value problem*. Higher-order linear ODEs can always be transformed into first-order linear ODEs. Note that if $A(t)$ is anti-Hermitian and $\bar{b}(t) = 0$, then Eq. (7.3) becomes the time-dependent Schrödinger equation, which is solved directly with Hamiltonian simulation. Equation (7.3) could be viewed within the framework of Eq. (7.1) with $d = 1$ and u a vector-valued rather than scalar-valued function. We separate these cases because the existing literature typically uses either Eq. (7.1) or Eq. (7.3) as its starting point, and the methods pursued in each case are distinct.

For nonlinear PDEs, the linear equations in Eq. (7.1) and Eq. (7.3) are replaced by nonlinear ones. For example, one can extend Eq. (7.3) to consider an ODE with a polynomial nonlinearity of the form

$$\frac{\mathrm{d}\bar{u}(t)}{\mathrm{d}t} = F_M(t)\bar{u}(t)^{\otimes M} + A(t)\bar{u}(t) + \bar{b}(t)\,, \tag{7.4}$$

where $F_M(t)$ is a tensor encoding the nonlinearity, although note that existing quantum algorithms have focused on the case where $F_M(t)$ and $A(t)$ are time independent. This class of differential equations includes important potential applications such as CFD, since the Navier–Stokes equation is nonlinear with a quadratic nonlinearity ($M = 2$).

What does it mean to "solve" the differential equation? In the most general sense, this refers to obtaining an expression for the solution $u(x)$ (for Eq. (7.1)) or $\bar{u}(t)$ (for Eq. (7.3)). While closed-form solutions can be derived for some simple differential equations, this is not possible in general, and the solution typically must be computed numerically. However, in specific applications, we often do not need complete information about the solution function $u(x)$ or $\bar{u}(t)$ to accomplish a certain goal. An end-to-end problem involving the solution of a differential equation boils down to estimating the value of some *property* of the solution, denoted by $\mathcal{P}[u] \in \mathbb{R}$, up to specified additive error parameter ϵ. For linear PDEs, a straightforward example is when the property $\mathcal{P}$ is simply the value of u at a specific point x^*, that is, $\mathcal{P}[u] = u(x^*)$. More generally, we restrict to the case where $\mathcal{P}[u]$ is a linear functional of u, that is, $\mathcal{P}[u] = \langle r, u\rangle := \int_{x\in\Omega} \mathrm{d}x\, r(x)u(x)$ for some subset $\Omega \subset \mathbb{R}^d$ and function $r : \Omega \to \mathbb{R}$ for which $\langle r, r\rangle = 1$ [777]. For example, in [295], a quantum algorithm for solving Maxwell's equations based on the FEM was given, where the quantity of interest was not the electric field itself at any specific point, but rather the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

electromagnetic scattering cross section. In this case, the cross section was given by the square of a linear functional of u.

Properties of ODEs can be treated in the same framework, where we are interested in computing quantities of the form $\mathcal{P}[u] = \int_{t\in\Omega} \mathrm{d}t\ \bar{r}^\intercal(t)\bar{u}(t)$, which are linear functionals of the entries of $\bar{u}(t)$ over some subset Ω of the interval $[0, T]$. However, we note that for initial value problems, often of primary interest are properties at the final time T, in which case $\mathcal{P}[u]$ would reduce to an inner product $\bar{r}^\intercal\bar{u}(T)$. For example, in [832], the drag force on a ship hull was expressed as a linear functional of the solution to the lattice Boltzmann equation evolved forward in time.

Dominant resource cost/complexity

There are many distinct approaches to constructing a quantum algorithm for solving the end-to-end problem above. The exact complexity will of course depend on the method, but it will also depend on specific details related to how the differential equation and its boundary conditions are specified to the quantum algorithm (input model), as well as instance-specific factors such as how smooth the solution to the differential equation is. Here we overview some of the available choices and complexity considerations, focusing the bulk of the discussion on methods that leverage the quantum linear system solver (QLSS) as a primitive, as these have received the most attention in the literature.

Discretization of linear PDEs: Any numerical method must perform some form of discretization. First, we focus on linear PDEs such as Eq. (7.2) where there is no time variable. The choice of discretization will depend sensitively on the problem at hand. In the case of the Poisson equation in Eq. (7.2) with Dirichlet boundary conditions, quantum algorithms leveraging discretization schemes based on FDM, FEM, and spectral methods were discussed in [228], [777], and [285], respectively. The key goal is to minimize the number N of grid points or basis functions while achieving discretization error $O(\epsilon)$. Using low-order grid-based methods, a problem in d spatial dimensions requires taking $N = (1/\epsilon)^{\Omega(d)}$ grid points, with some caveats on solution norm and regularity [777]. Alternative sparse-grid or spectral methods can improve the $1/\epsilon$ dependence to logarithmic, but still scale exponentially with d [285]; however, these generally require stricter regularity requirements on the solution to the differential equation, which may not be satisfied in applications.

After appropriate discretization, the linear differential equation in Eq. (7.1) (along with its boundary conditions) reduces to a matrix equation:

$$L|u\rangle = |f\rangle\,. \tag{7.5}$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

This is the same linear equation that would be obtained for a classical method using the same discretization scheme. Information about the solution function $u(x)$ is encoded into the N components of the vector $|u\rangle$.[1] Classical methods typically manipulate a full description of all components of the vector $|u\rangle$, whereas quantum methods can create the normalized quantum state $|u\rangle/\||u\rangle\|$ encoding these N components into its amplitudes with $O(\log(N))$ qubits.

If the linear PDE has a time variable, one option is to treat it equally as the other $d-1$ variables (see, e.g., [692]), but it is often treated separately. First, discretization of the other $d-1$ variables using N total grid points or basis functions is performed as above, which reduces the linear PDE in Eq. (7.1) to an ODE with N variables as in Eq. (7.3). Time is then discretized and propagated as discussed for ODEs below.

Discretization of time in linear ODEs: To solve the linear ODE on N variables in Eq. (7.3)—whether it came about via discretization of a PDE or otherwise—the time interval $[0, T]$ is discretized into grid points, and the solution at one grid point is related to the solution at the prior grid point by a time-ordered exponential of the matrix A. If A is time independent, this exponential can be approximated by a truncated Taylor series [138, 646, 571], and if A is time dependent, it can be approximated by a truncated Dyson series [132]. The number of grid points needed scales linearly with T, and the series is truncated at order $\text{polylog}(T/\epsilon)$. An alternative approach when $\bar{u}(t)$ is sufficiently smooth in time uses spectral methods, which approximate $\bar{u}(t)$ as a truncated series over a complete set of basis functions [279]. In any case, the relation between the solution at different time grid points or basis functions leads to a linear system of equations, now of size roughly $N' = O(NT)$, but again of the form $L|u\rangle = |f\rangle$ as in Eq. (7.5).

Here the solution $|u\rangle$ is a "history state" meaning that it is given by a *superposition* of states $|t\rangle|\bar{u}(t)\rangle$ for different discrete values of t across the entire interval $[0, T]$. Since one is often interested only in $\bar{u}(T)$, additional trivial time steps can be included at the end to boost the portion of the history state amplitude on the $t = T$ branch [131].

It is important to emphasize that classical methods for solving ODEs do not solve the same linear equation $L|u\rangle = |f\rangle$ arrived at by these methods. Rather, they typically handle time in a time-marching fashion where the value of $\bar{u}(t)$ at one time step is directly computed from its value at one or more previous time steps. In [379], a quantum time-marching strategy was proposed

[1] In this chapter, we adopt a convention where quantum states like $|u\rangle$ need not be normalized states. In fact, the norm, denoted by $\||u\rangle\|$ will be important for reasoning about the complexity.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

for time-dependent homogeneous linear ODEs, which generates a sequence of quantum states $|\bar{u}(0)\rangle, |\bar{u}(t_1)\rangle, |\bar{u}(t_2)\rangle, \ldots, |\bar{u}(T)\rangle$ (rather than a superposition of these states). This method avoids the need to solve a linear system, but it does utilize primitives from quantum linear algebra. Similarly, the methods of [312, 579, 80, 581, 36, 37] avoid the need to solve linear systems by mapping ODEs to equations that can be simulated with Hamiltonian simulation [312, 579, 80, 581], or linear combination of Hamiltonian simulation [36, 37].

Solving the linear system: Once the linear PDE or ODE has been reduced to a linear system $L|u\rangle = |f\rangle$, it can be solved on a quantum computer by applying the QLSS. The QLSS subroutine prepares a quantum state approximating the normalized solution vector $|u\rangle/\||u\rangle\|$ up to some specified precision ξ in ℓ_2 norm, where $\||u\rangle\| = \sqrt{\langle u|u\rangle}$ is the norm of the quantum state encoding the solution to the linear system. To do so, the QLSS assumes access to oracles that (coherently) query the matrix elements of L and prepare the normalized state $|f\rangle/\||f\rangle\|$. Optimal QLSSs [313, 327] (see also alternative near-optimal methods in [26, 282, 248, 964, 31, 571]) make $O(s\kappa \log(1/\xi))$ queries to these oracles, where κ is the condition number of the matrix L (i.e., the ratio of the largest and smallest singular values), and s is the maximum number of nonzero elements in any row or column of L ("sparsity"). Additionally, one can compute an estimate for the norm $\||u\rangle\|$ up to relative error ξ using $\widetilde{O}(s\kappa/\xi)$ queries (note the worse ξ-dependence) [327, 248]. For simplicity, we assume that to achieve ϵ overall error on the end-to-end problem, it will suffice to take $\xi = O(\epsilon)$, although there can also be factors that depend on the choice of discretization and norms of the solution u (see, e.g., [777]).

Henceforth, let N' refer to the size of the linear system being solved, so $N' = N$ for the PDE example described above, and $N' = O(NT)$ for the ODE example (with N reserved for the size of the matrix A).

The oracles for querying the matrix elements of the s-sparse $N' \times N'$ matrix L and for preparing the N'-dimensional state $|f\rangle/\||f\rangle\|$ are assumed to have cost polylog(N'). This is valid if the matrix elements of L can be efficiently computed "on the fly," which is plausible when they are given by succinct expressions, for example, based on a simple finite difference formula. However, if entries of L and $|f\rangle/\||f\rangle\|$ depend on arbitrary, classically stored data related to, for instance, object geometries, boundary conditions, or grid point locations, then the assumption of polylog(N') cost per query requires access to a log-depth quantum random access memory (QRAM). This requirement is necessary in many practical applications of PDEs involving highly complex geometries in three spatial dimensions, such as CFD and seismic modeling. The assumption of log-depth QRAM brings significant caveats—for example,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

while the quantum circuits for implementing the QRAM operation can be parallelized to have depth polylog(N), they cannot avoid having size poly(N) for accessing a database with poly(N) entries; see Chapter 17 on loading classical data for more information.

With these assumptions, the QLSS portion of the quantum algorithm can be performed exponentially faster in the parameter N, and with exponential saving in memory, compared to any classical method that manipulates vectors of size N, which includes Gaussian elimination and iterative methods like conjugate gradient. Specifically, the quantum complexity to obtain the state $|u\rangle/\||u\rangle\|$ is given by

$$s\kappa \cdot \text{polylog}(N', 1/\epsilon)$$

and the cost to obtain $\||u\rangle\|$ is $s\kappa\epsilon^{-1} \cdot \text{polylog}(N', 1/\epsilon)$. The number of qubits is $O(\log(N'))$, although if a log-depth QRAM is necessary, this may require poly(N') ancilla qubits.

Reading out estimates for properties of the solution to the differential equation: Preparing the $\log(N')$-qubit state $|u\rangle/\||u\rangle\|$ does not immediately yield an estimate for the property $\mathcal{P}[u]$. Indeed, reading out useful information from $|u\rangle/\||u\rangle\|$ represents a major bottleneck of the algorithm. Consider the case that $\mathcal{P}[u]$ corresponds to the value $u(x^*)$ at a specific point x^* (for PDEs), or the amplitude $\langle x^*|\bar{u}(T)\rangle$ on one of the basis states (for ODEs). Then, the estimation of $\mathcal{P}[u]$ to precision ϵ is performed with amplitude estimation, which introduces multiplicative overhead $O(\||u\rangle\|/\epsilon)$ into the complexity. To read out all N amplitudes of the state $|u\rangle$ in this fashion, a linear factor of N would be reintroduced, although more advanced methods of pure state tomography can reduce this to $\sqrt{N}$ [49]. In the more general case that $\mathcal{P}[u]$ is a linear functional, the value of $\mathcal{P}[u]$ can be expressed (up to discretization error) as an overlap $\mathcal{P}[u] = \langle r|u\rangle$ between some preparable normalized state $|r\rangle$ and the solution vector $|u\rangle$ that solves $L|u\rangle = |f\rangle$. Thus, to compute $\mathcal{P}[u]$, one computes the overlap between $|r\rangle$ and $|u\rangle/\||u\rangle\|$, and then multiplies by $\||u\rangle\|$. Overlap estimation [637] is a straightforward application of amplitude estimation, and achieving precision $\epsilon/\||u\rangle\|$ introduces $O(\||u\rangle\|/\epsilon)$ multiplicative overhead. Thus, the overall scaling of the complexity is

$$\frac{s\kappa\,\||u\rangle\|}{\epsilon} \cdot \text{polylog}(N', 1/\epsilon)\,. \tag{7.6}$$

For initial value problems governed by the ODE in Eq. (7.3), one is often interested in properties $\mathcal{P}[u]$ that depend only on the solution $\bar{u}(T)$ at the final time T. When $|u\rangle = \sum_i |t_i\rangle|\bar{u}(t_i)\rangle$ is a history state encoding of the solution $\bar{u}(t)$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

over the whole interval, the complexity of computing useful information will grow with the ratio

$$q = \frac{\sup_{t\in[0,1]} \| \, |\bar{u}(t)\rangle \, \|}{\| \, |\bar{u}(T)\rangle \, \|} \approx O\left(\frac{\| \, |u\rangle \, \|}{\| \, |\bar{u}(T)\rangle \, \|}\right),$$

that is, the factor by which the norm of the solution has decayed compared to its maximum on the interval $[0, T]$ (the approximation is correct assuming $\sup_{t\in[0,T]} \| \, |\bar{u}(t)\rangle \, \|$ accounts for a constant fraction of the total norm $\| \, |u\rangle \, \|$). This arises from the fact that $|T\rangle|\bar{u}(T)\rangle$ contributes at most $\| \, |\bar{u}(T)\rangle \, \| \approx O(\| \, |u\rangle \, \|/q)$ amplitude to the history state $|u\rangle$ and thus the additive precision ϵ will need to be on the order of $\| \, |u\rangle \, \|/q$, or smaller, to learn something useful about $\bar{u}(T)$. Since the complexity scales linearly with $\| \, |u\rangle \, \|/\epsilon$, the complexity grows with q. Unfortunately, q can be large and growing with T if the solution to the ODE is decaying. Furthermore, the dependence of the complexity on q is necessary since otherwise the algorithm would be able to perform postselection on low-probability events, a power ruled out by widely believed complexity-theoretic conjectures; see [138, 35].

The persisting polylog(N) dependence in Eq. (7.6) suggests an exponential speedup in the parameter N compared to classical methods, but this conclusion depends on the scaling of the parameters s, κ, and $\| \, |u\rangle \, \|/\epsilon$ with N.

Condition number: The sparsity s and condition number κ depend on the differential equation and the choice of discretization, but can often be controlled. For example, for PDEs discretized by the FEM, in many instances we have $s = O(1)$ and $\kappa \leq O(N^{2/d})$ (see, e.g., [198, Theorem 9.7.1]). Additionally, preconditioning of linear systems to reduce κ is an effective technique in classical iterative approaches to solving linear systems such as the conjugate gradient method, and several studies have examined the possibility of integrating these into the QLSS in some circumstances [295, 925, 990, 83]. In the best case scenario, these could reduce κ to $O(1)$.

In the setting of ODEs like Eq. (7.3), upper bounds on κ can be derived. These bounds can have the form $\kappa \leq CT$, where C is a factor that depends on the spectral properties of A.[2] The upper bounds on κ also require an assumption that the ODE is "dissipative" [131, 138, 646, 132]; otherwise, the value of C in the bound can grow exponentially with T [646], consistent with the observation that the norm of the history state $|u\rangle$ can be exponentially larger than the norm of the initial state $|\bar{u}(0)\rangle$. A sufficient condition for the ODE to be dissipative is that A is diagonalizable and the real parts of its eigenvalues

[2] Generally, the bound on κ scales with the number of grid points in time. The linear-in-T bound is achieved when using the Dyson series method with high-order truncation [132].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

are non-positive [138] so that the solutions are stable and do not grow with time, although this technical definition was relaxed slightly in [646] and now includes nondiagonalizable A. The requirement of dissipation is not as relevant for classical solvers based on time-marching strategies, which can renormalize growing solutions at each step and do not generally require solving linear systems.

The linear dependence on T in the complexity cannot be improved in general since this factor appears in the complexity of optimal Hamiltonian simulation, which corresponds to the special case that A is anti-Hermitian and $\bar{b} = 0$ [134]. However, it has been shown that many ODEs admit "fast forwarding" and the dependence on T can be reduced. For example, for stable ODEs (when all eigenvalues of A have a negative real part), a bound of the form $\kappa \leq \widetilde{O}(\sqrt{T})$ was derived in [571]; see also [35].

Final complexity: For dissipative ODEs on N variables, propagated to time T, the final complexity inherits a linear dependence on CT via the condition number, and existing literature typically includes the factor q defined above directly in the complexity statements. These statements are phrased to say that the state $|\bar{u}(T)\rangle$ can be obtained to error ξ in complexity roughly $TsqC \cdot \mathrm{polylog}(N, 1/\xi)$ [138, 646, 571, 132], which accounts for postselecting the time register to $t = T$ but not yet the complexity to read out a property of interest.[3] Defining $\epsilon' = \epsilon q / \| |u\rangle \|$ to be the normalized precision parameter that should be small in order to learn something interesting, and taking $\xi = O(\epsilon')$, the total end-to-end complexity including readout could then be expressed as

$$\frac{TsqC}{\epsilon'} \cdot \mathrm{polylog}(N, q/\| |u\rangle \| \epsilon') \,. \tag{7.7}$$

For PDEs in d dimensions, we recall that N and ϵ are not independent parameters: in general, we are interested in simulating the PDE to a fixed precision, and adapt N to reach the desired precision. As discussed, depending on the discretization method, N scales either as $(1/\epsilon)^{O(d)}$ or $(\mathrm{polylog}(1/\epsilon))^d$, but either way, we have that $\mathrm{polylog}(N) \leq d^{O(1)} \cdot \mathrm{polylog}(1/\epsilon)$. For PDEs with a time variable and initial conditions specified at $t = 0$, where $d - 1$ dimensions are discretized and time is integrated via mapping to an ODE, we substitute

[3] We briefly mention the complexity of alternative approaches to ODEs that avoid solving linear systems. The quantum time-marching method of [379] has a different complexity form, but has similar features, growing with time (in this case, as T^2) and depending on an "amplification ratio" $Q > q$, but offering other potential benefits, such as minimal regularity requirements (even allowing $A(t)$ to have jump discontinuities) and needing fewer queries to the initial condition $\bar{u}(0)$. Meanwhile, the linear combination of Hamiltonian simulation method [37] shares the feature of needing minimal queries to $\bar{u}(0)$ while matching the complexity stated above.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

this value of N into Eq. (7.7). This gives a total complexity of

$$d^{O(1)} \cdot \widetilde{O}(TCq/\epsilon')$$

for reading out a property of the (renormalized) solution to the PDE at time T to precision ϵ'.

For static PDEs where all d dimensions are discretized by a grid-based method like the FEM, we instead substitute $\kappa = O(N^{2/d})$, $s = O(1)$, and $N = (1/\epsilon)^{O(d)}$ into Eq. (7.6), yielding overall end-to-end complexity

$$\frac{\| |u\rangle \| \, d^{O(1)}}{\epsilon^{1+O(1)}}$$

to compute a global property of the PDE, given conditions on the boundary. If preconditioning improves κ to $O(1)$, the ϵ dependence is improved to essentially linear in $1/\epsilon$—see [777] for a more careful analysis specific to the FEM that arrives at similar expressions, but better accounts for impact of solution norm and smoothness.

Generally, the main conclusion is that—irrespective of the discretization scheme—the quantum complexity is polynomial in the desired precision $1/\epsilon$, although for d-dimensional PDEs the complexity may scale as $\text{poly}(d) \cdot \text{poly}(1/\epsilon)$ rather than $\text{poly}((1/\epsilon)^d)$. Thus, for fixed dimension d there is potentially room for a polynomial-in-$1/\epsilon$ quantum speedup, the size of which grows exponentially in the dimension d. Ultimately, the necessity of the $O(1/\epsilon)$ scaling is traced back to the fact that the quantum solver produces a quantum state encoding the normalized solution to the differential equation, potentially exponentially faster than leading classical methods such as conjugate gradient, but the exponential speedup is lost in the readout step, where one must learn an observable of interest to error ϵ. Moreover, this conclusion holds not just for "bad" observables (like full state tomography), but for any observable, due to the $\Omega(1/\epsilon)$ cost of quantum readout.

Nonlinear differential equations: The immediate issue with applying the above methods to nonlinear differential equations such as the nonlinear ODE of Eq. (7.4) is that discretization no longer leads to a system of linear equations. Early work on quantum algorithms for nonlinear ODEs handled this issue by dividing time into short time steps and preparing the quantum state encoding the solution at one time step using multiple copies of the solution at the previous time step [679]. Since quantum states cannot be cloned, the complexity of this strategy necessarily grows exponentially in the number of time steps. More recent quantum algorithms for nonlinear differential equations instead proceed by first linearizing the differential equation and then applying

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the methods sketched above [710, 702, 1061, 703, 646, 315]. Specifically, the most heavily studied approach has been Carleman linearization, where one exactly maps a nonlinear ODE with polynomial nonlinearity such as Eq. (7.4) to a linear ODE on an infinite-dimensional variable $(\bar{u}, \bar{u}^{\otimes 2}, \bar{u}^{\otimes 3}, \ldots)$, and then truncates to form an approximate finite-dimensional linear ODE. Quantum algorithms based on this method were first studied in [702] and further developed in [703, 646, 315, 207], and it has also been analyzed in the context of specific differential equations such as reaction-diffusion equations [703, 315, 32], the Navier–Stokes equation (via the lattice Boltzmann equation) [683, 832], and differential equations related to training classical neural networks [701]. The complexity of the quantum algorithm has a similar scaling as that for linear ODEs quoted in Eq. (7.7), growing linearly in T, q, and $1/\epsilon'$. However, this complexity scaling requires an additional assumption: a quantity R capturing the nonlinearity-to-dissipation ratio of the differential equation must satisfy $R < 1$ for the errors to be controlled (see [702, 703, 315, 1055]), and it is not always clear when this condition holds.

For example, in the case of the Navier–Stokes equation, the size of the nonlinearity—and hence the value of R—grows with the "Reynolds number" of the fluid, and the condition $R < 1$ would be violated when simulating high-Reynolds-number turbulent flows. Turbulent flows are potentially handled by applying the Carleman linearization method to the lattice Boltzmann equation rather than the Navier–Stokes equation directly [683, 832], in which case the size of the nonlinearity does not scale with the Reynolds number. Indeed, generally speaking, for this approach to nonlinear differential equations, there is a delicate interplay between the size of the input state at time $t = 0$, the form of the nonlinearity, and the amount of dissipation in the linear term; see [315] for a discussion.

Separate from these approaches, methods have been proposed that map nonlinear classical dynamics to linear phase-space dynamics that can be simulated with Hamiltonian simulation [590, 354, 580, 578, 581].

Comments on the complexity of alternative methods and problems: We briefly comment on two further classes of applications involving PDEs, but which typically have very different characteristics. The first is stochastic differential equations (SDEs), which are simulated extensively in computational finance and more generally in risk modeling. There, one typically samples trajectories of the SDE (via Monte Carlo methods), and evaluates observables stochastically. Quantum-accelerated Monte Carlo has been worked on extensively (see Section 8.2 on pricing financial options). Where classical methods require sampling $O(1/\epsilon^2)$ trajectories to obtain an ϵ-estimate of a certain quan-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tity, the quantum method can create a superposition of trajectories and read out the relevant amplitude at $O(1/\epsilon)$ cost—a quadratic quantum advantage. On the classical side, a key advantage of these methods is that they avoid an exponential scaling in the number of continuous dimensions d (i.e., the "curse of dimensionality"), unlike the "Eulerian" approaches discussed above that discretize the d-dimensional space into $N \geq \exp(\Omega(d))$ grid points or basis functions. Thus, they are relatively effective when d is large, and less preferred when d is small due the fact that their ϵ dependence cannot be better than $O(1/\epsilon^2)$. In fact, in some cases, classical and quantum trajectory-based methods can be applied not only to SDEs but also to (deterministic) PDEs, and thus they compete against QLSS-based PDE and ODE methods we have discussed. For example, the quantum and classical complexity of a Monte Carlo approach for the heat equation was studied in [692] and compared against alternative approaches—for the end-to-end problem considered, the classical Monte Carlo approach outperformed all classical alternatives when $d > 4$, and the quantum-accelerated Monte Carlo approach outperformed all (quantum or classical) alternatives when $d > 2$. For SDEs, an alternative to Monte Carlo is to map the SDE to a Fokker–Planck equation via the Itô calculus and solve the Fokker–Planck PDE. This has been proposed in [441]. However, for most SDEs of interest in risk analysis, Monte Carlo simulation converges in a number of samples scaling linearly in the number of variables, leaving very little room for a quantum speedup in these applications given our current understanding.

The last class of problems to be mentioned are multi-particle Schrödinger equations. They are (i) high dimensional, (ii) complex, and (iii) require high-precision solutions for practical applications. Hence, they match all of the criteria under which a quantum advantage might be expected. The second-quantized approach to solving the full configuration interaction molecular Schrödinger equation is a specific case of the spectral method, although here one must solve an eigenvalue equation rather than a linear system. Unsurprisingly, this case has already gathered a lot of attention (see Chapter 2 on quantum chemistry).

Existing resource estimates

An explicit resource estimate for linear PDEs was reported in [902] for solving Maxwell's equations to estimate an electromagnetic scattering cross section in 2D. Following the asymptotic analysis of [295], it employed an FEM-based discretization scheme to form a linear system of size $N = 3 \times 10^8$, targeting accuracy $\epsilon = 0.01$. The estimates did not incorporate preconditioning methods and assumed a value for the condition number $\kappa \approx 10^4$, ultimately finding

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

that 10^{29} T gates would be needed to complete the computation. However, this work predated asymptotic and practical advancements to the complexity of the QLSS [313, 327], and modern estimates for the same problem would likely lead to more reasonable resource counts. Note also that much of the art in classical PDE solvers is to find appropriate preconditioning schemes to control the condition number. In [295], it was shown that one common class of preconditioners works within the framework of the quantum algorithm, but it is as yet unclear if this is the case more generally.

For ODEs, [571] gave a detailed performance analysis of the Taylor series truncation approach developed in [138, 646] applied to general time-independent dissipative ODEs. They gave explicit upper bounds on the condition number κ of the linear system in terms of the total evolution time T and the "log-norm" of the matrix A. They considered the task of outputting the history state $|u\rangle$ or the final-time state $|\bar{u}(T)\rangle$. By combining the bound on κ with explicit upper bounds from [572] on the query complexity of the QLSS, they determined an upper bound on the number of times the algorithm needs to query a block-encoding of the ODE matrix A to accomplish this task. The estimated number of queries per unit time varied from 10 to 10^5 over the parameter regime considered, and these figures would be reduced with subsequent improvements to the QLSS complexity, such as those reported in [327].

In [832], the query bounds of [571] were applied to the specific end-to-end CFD problem of computing the drag force on a ship hull in the incompressible (and potentially turbulent) parameter regime, by solving the nonlinear lattice Boltzmann equation (linearized via Carleman linearization). They considered a simplified version of the problem where the ship hull is modeled as a sphere. They estimated that the quantum algorithm would need 10^{20}–10^{24} T gates and roughly 10^3–10^5 logical qubits (depending on the value of the Reynolds number) to compute the drag force on the sphere. Classically, the lattice Boltzmann method is a high-accuracy method and would be computationally intractable for this instance in the high-Reynolds-number regime. In practice, classical methods resort to lower accuracy methods, which for this instance can be completed within several minutes on a laptop. Computing the drag force on actual ship hulls with the quantum method is expected to be significantly more resource intensive compared to the flow-past-a-sphere instance due to the need to coherently load the boundary conditions describing the ship hull each time the block-encoding is queried.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Caveats

A key caveat is that many analyses in the literature do not consider the full end-to-end problem that needs to be solved for applications. Often, these works only consider the cost of preparing the quantum state $|u\rangle$ that encodes the solution to the differential equation into its amplitudes, and they report this cost in terms of the number of queries to oracles of the input data. Thus, these works study the task of solving differential equations as a primitive—similar to Hamiltonian simulation, that is, simulation of the time-dependent Schrödinger ODE—rather than as an end-to-end application. As discussed above, reading out useful information to precision ϵ introduces a $\Omega(1/\epsilon)$ multiplicative overhead and dramatically changes the complexity scaling, precluding exponential end-to-end speedups. Furthermore, whereas a full classical description of the solution could be computed just once, and subsequently many properties read out from that description, the state $|u\rangle$ is consumed during readout, and the number of times $|u\rangle$ needs to be prepared grows with the number of properties one wants to learn. In some cases, one may only seek to learn a few observables, but in other cases, extracting the desired information might require near full tomography of the quantum state $|u\rangle$, which in certain situations removes all quantum advantage [692].

The readout caveat can potentially be avoided if a small number of *samples* from the state $|u\rangle$ measured in the computational basis, as opposed to properties $\mathcal{P}[u]$ as defined above, would be useful for the end-to-end application. However, in such cases one must also be careful to compare against classical methods for the same task, where quantum-inspired methods can be competitive [977]. In [80], it was shown that BQP-hard problems can be encoded into an ODE describing coupled oscillators and solved by sampling from $|u\rangle$, but this situation would be unlikely to arise naturally in applications. In [701], it was suggested that samples from $|u\rangle$ encoding the solution to certain nonlinear differential equations could be useful for training neural networks.

Another caveat is that complexity statements often report only the number of times the algorithm queries oracles for the input data. In applications where the input data encoding complex boundary conditions or object geometries is not efficiently computable but rather stored in a large classical database, one must assume access to a log-depth QRAM in order to implement these oracles efficiently, an assumption that has its own caveats.

For simulating time evolution of ODEs, it is important to emphasize the dependence of the complexity in Eq. (7.7) on the parameter q, which for dissipative systems grows with time, potentially exponentially, as the size of the solution decays. Furthermore, for nonlinear differential equations, we reiterate

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

that existing quantum algorithms are often based on the assumption that the nonlinearity-to-dissipation ratio is sufficiently small, which may not be satisfied in practical instances. Generally speaking, strong nonlinearities can cause the solution to develop discontinuities, and linearization schemes might break down for problems of interest if the solutions lack sufficient regularity, as can be the case for simulations of turbulence in CFD.

Finally, we note that due to the large number of methods available to classical solvers of differential equations, an important caveat is that any claim of quantum advantage must be sure to compare against the best possible classical method, and consider the possibility that this classical method might benefit from parallelization.

Comparable classical complexity and challenging instance sizes

Classical algorithms for linear PDEs can compute a classical description of the solution u by solving the same linear equation $L|u\rangle = |f\rangle$ solved by the quantum algorithm. For an s-sparse $N \times N$ linear system, the complexity of an exact Gaussian elimination approach is $O(N^\omega)$, where $\omega < 2.37$ is the matrix multiplication exponent. However, in practice, approximate iterative methods such as the conjugate gradient method are more common. The complexity of conjugate gradient scales as $\widetilde{O}(Ns\sqrt{\kappa}\log(1/\epsilon))$ when L is positive semidefinite [481, Chapter 10.2]. For a discretization scheme like the FEM where $N = (1/\epsilon)^{\Omega(d)}$, and using the aforementioned bound $\kappa \leq O(N^{2/d})$, the complexity of the conjugate gradient approach is $s(1/\epsilon)^{\Omega(d)}$, which has exponential dependence on the spatial dimension d but for fixed d scales as $\text{poly}(1/\epsilon)$. Additionally, in practice, the conjugate gradient method benefits from preconditioning techniques to reduce κ, and from parallel implementation on graphics processing units (GPUs). For a sense of scale, [713] used the preconditioned conjugate gradient method within a finite element analysis to compute the thermal conductivity and elasticity of certain 3D cast iron samples imaged with micro-computed tomography (a task chosen mainly to benchmark their method). Among other reported results, their implementation solved the end-to-end problem, which required solving several linear systems, with $N \geq 10^6$ in less than 1 second, and $N \geq 4 \times 10^8$ in less than 30 minutes using a single GPU with 8 gigabytes of RAM.

For linear and nonlinear initial value problems, classical methods could apply conjugate gradient or other linear system solvers to the same linear equation $L|u\rangle = |f\rangle$ that the quantum algorithm constructs to solve the ODE. The complexity of this approach would have a linear-in-N dependence, but since the solution would be a full classical description of the history state, it would

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

not need to also pay the factor of q arising from postselecting on the $t = T$ branch of the history state, and it would avoid the $O(1/\epsilon)$ readout cost.

However, most classical methods do not follow this route and instead integrate the ODE with a time-marching method, where a description of the length-N vector is propagated forward in time, for which there are many options [1024, 1025]. Similar to the quantum complexity, nearly linear-in-T scaling is achieved by high-order methods, so long as $A(t)$ is smooth up to corresponding order, or by spectral methods [930]. In the time-independent or smooth case, one would achieve $NTC' \cdot \text{polylog}(T, 1/\epsilon)$ complexity, where C' is some constant depending on the spectral properties of A, similar to C. As in the quantum case, care must be taken to choose ϵ appropriately when solutions are exponentially growing or decaying.

We also mention that there has recently been work on using machine learning methods to classically simulate nonlinear PDEs, especially for CFD [1019, 730]. These methods are generally very fast, but they are heuristic, so they are suitable in some instances but not when high-confidence, high-accuracy solutions are required.

Speedup

For linear PDEs in d dimensions, solved via discretization with $N = (1/\epsilon)^{\Omega(d)}$ grid points and inverting the corresponding linear system, the speedup is a reduction from time roughly $\epsilon^{-\Omega(d)}$ classically to $d^{O(1)}\epsilon^{-1}\epsilon^{-O(1)}$ quantumly, here omitting dependencies on $\| |u\rangle \|$ and $\log(\epsilon^{-1})$. The $O(1)$ powers depend on the details of the discretization and the efficacy of preconditioning. Other discretization schemes may give rise to slightly distinct complexity forms, but in any case, the conclusion is that for fixed dimension d, the speedup is at best *polynomial*, a point that has been made in more detail in, for example, [777, 692].

The speedup can be exponential in the parameter d. However, in many engineering applications, the number of dimensions is fixed to be fewer than four (three for space, one for time), limiting the advantage quantum methods can obtain. Furthermore, for PDEs with large d, trajectory-based classical strategies can avoid the exponential-in-d complexity scaling, and in cases where these methods apply, the best possible speedup is typically a quadratic reduction from $O(1/\epsilon^2)$ to $O(1/\epsilon)$; see, for example, [692] and Section 8.2 on option pricing.

For integrating ODEs of N variables forward in time, there can be an exponential speedup in the parameter N. However, since N and ϵ are typically related by $N \leq (1/\epsilon)^{\Omega(d)}$ (e.g., when the ODE arises via discretization of a PDE), the $O(1/\epsilon)$ cost of readout will contribute a much larger factor than the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

polylog(N) cost of the QLSS, and the best possible speedup is again polynomial. An exponential speedup could be possible if $N \geq \exp(\Omega(1/\epsilon))$, or if samples from the state $|u\rangle$ were directly useful within the end-to-end application, but this assessment would also require that the solution-decay-factor q appearing in the quantum complexity does not cancel the speedup.

In general, these methods do offer the possibility of an exponential saving in space, since the quantum methods can represent the vector using a logarithmic number of qubits. Nevertheless, the overall take-home message is that quantum algorithms can potentially outperform classical algorithms, but major gains are only to be expected when the number of spatial dimensions is large, or if there is otherwise a reason that the linear systems involved are much larger than the precision demanded in the output. This intuition is corroborated by the analysis of quantum computing algorithms for ab initio chemistry, where the number of dimensions scales with the number of electrons.

NISQ implementations

Various proposals at NISQ implementations of PDE solvers have been made; see [677] and references therein. The idea is to start from some discretization of the PDE $L|\psi(\theta)\rangle = |b\rangle$, where $|\psi(\theta)\rangle$ is an appropriately chosen variational circuit, and to optimize the parameters θ of the circuit. This is an example of a variational quantum algorithm. Another proposal applies a variational approach to nonlinear PDEs [725]. Note that even if these methods find parameters to generate a good approximation of the solution, they would still pay the $O(1/\epsilon)$ cost to read out properties. Thus, they offer at best a polynomial speedup over classical methods. It is difficult to imagine that sufficient size and precision can be reached in the NISQ regime to be competitive with the best classical solvers.

Outlook

While the simulation of differential equations is one of the most important large-scale computational tasks, constituting a sizable fraction of HPC workloads in industry, at present the benefit of quantum solvers for real-world problems appears limited to relatively modest polynomial speedups. Extensive work on quantum algorithms for solving differential equations has developed methods with rigorous analyses and likely close-to-optimal complexities; the challenge that remains is how these methods can integrate into an end-to-end application pipeline, in such a way as to reduce the cost. To find a high-value application related to solving differential equations (beyond potentially ab initio chemistry), one would likely need to find a situation that meets some or all of the following criteria: (i) involves a very large number of spatial dimensions,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

(ii) has simple geometry or initial conditions in order to avoid the need for a QRAM input model, (iii) requires high precision, ruling out heuristic classical approximate methods, (iv) requires learning a relatively small number of properties of (or ideally requires only samples from) the solution vector. There remains the possibility for substantial improvements in memory usage, but this is not currently a bottleneck in classical PDE solving.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

8

Finance

The financial services industry is among those beginning to explore the potential future benefits of quantum computing. Finance has the distinct feature that more powerful and more accurate simulations can lead to direct competitive advantage, in a way that is harder to identify in other industries. In this application area, researchers strive to find quantum speedups for use cases of interest to financial services. A number of use cases have been proposed as candidates for quantum solutions, such as:

- **Derivative pricing** (such as options [949, 867], and collateralized debt obligations (CDOs) [981]): Derivatives are financial instruments that are built upon an underlying asset (or assets) that can depend on the value of the asset in potentially complicated ways. In the derivative pricing problem, one needs to determine a fair price of the financial instrument, which is the price that would be received by the seller or paid by the purchaser when an asset or liability is transferred between market participants in an orderly transaction. Typically, one needs to compute the expected value of the fair purchase price of underlying assets at some later date when pricing a derivative. A similar and related problem is known as *computing the Greeks* [950]. The Greeks of a financial derivative are quantities that determine the sensitivity of the derivative to various parameters in the problem. For example, the Greeks of an option are given by the derivative of the price of the option with respect to some parameter, for example, $\Delta := \partial V/\partial X$, where V is the price of the option and X is the price of the underlying asset.
- **Credit valuation adjustments (CVAs)** [491]: CVA is the problem of determining the fair price of a derivative, portfolio, or other financial instrument while taking into account the purchaser's (potentially poor) credit rating, and the risk of default. CVA is typically given by the difference between the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

default risk-free portfolio and the value of the portfolio taking into account the possibility of default.

- **Value at risk (VaR)** [1049]: Many forms of risk analysis can be considered, with VaR being a common example. VaR measures the total value a financial instrument (such as a portfolio) might lose over a predefined time interval within a fixed confidence interval. For example, the VaR of a portfolio might indicate that, with 95% probability, the portfolio will not lose more than $\$Y$. A similar technique works as well for the related conditional value at risk (CVaR) problem.
- **Portfolio optimization** [865]: The goal of portfolio optimization is to determine the optimal allocation of funds into a universe of investable assets such that the resulting portfolio maximizes returns and minimizes risk, while also respecting other constraints.

While there are are many more use cases and several approaches for generating quantum speedups, broadly speaking, many use cases stem from one of two paths to quantum improvements: quantum enhancements to Monte Carlo methods (for simulating stochastic processes), and constrained optimization. In the first case, the approach generally involves encoding a relevant, problem-specific function into a quantum state, and then using quantum amplitude estimation to sample from the distribution quadratically fewer times than classical Monte Carlo methods [773]. In the second case, a financial use case is reduced to a constrained optimization problem, and a quantum algorithm for optimization is used to solve the problem.

Among the use cases studied in these two areas, option pricing and portfolio optimization often serve as archetypal examples of Monte Carlo and constrained optimization problems, respectively, and their associated quantum algorithms have the most follow-up work. Moreover, these two classes of problems comprise a considerable fraction of the classical compute used in the financial services industry. For these reasons, we will focus on these two use cases in this chapter, though the approaches, caveats, and complexities typically translate to other relevant use cases.

In addition to the use cases described above, other areas of interest to the financial services industry include post-quantum cryptography, as well as quantum-secure networking and quantum key distribution. However, many of these topics or their proposed quantum implementations are outside the scope of this discussion. Quantum machine learning is yet another popular use case within quantum approaches to finance, but oftentimes these results are quantum approaches to standard machine learning problems, which are then applied to a financial application. As such, we will also not study machine

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

learning in this finance-specific chapter and instead refer interested readers to any of the excellent review articles on quantum finance (e.g., [525, 176]) for more details.

The authors are grateful to Nikitas Stamatopoulos for reviewing this section of the survey, to Patrick Rebentrost for reviewing Section 8.1, and to Ashley Montanaro for reviewing Section 8.2.

8.1 Portfolio optimization

Overview

Given a set of possible assets into which one can invest, the problem of portfolio optimization (PO) involves finding the optimal allocation of funds in these assets so as to maximize returns while minimizing risk. The Markowitz model, as it is commonly called, is widely used in the financial industry, owing to its simplicity and broad applicability. Sophisticated constraints, transaction cost functions, and modifications to the problem can be used to model realistic, modern PO problems. Numerically solving these optimization problems is a routine part of existing workflows in financial services operations. Several quantum approaches to solving the PO problem have been proposed, each with their own advantages and drawbacks.

Actual end-to-end problem(s) solved

Consider a set of n investable assets with a fixed total budget. Define $w_i \in \mathbb{R}$ to be the fraction of the total budget that is invested into asset i. Thus, the n-dimensional vector w defines a portfolio. Let r be a known n-dimensional vector denoting the expected return for each of the available assets, that is, the percentage by which the value of each asset is expected to grow over some defined time period. Let $\Sigma \in \mathbb{R}^{n \times n}$ be the covariance matrix governing the random (and possibly correlated) fluctuations in the asset returns away from their mean r. In practice, the input parameters Σ and r can be inferred from historical stock price data, or through more sophisticated analyses. The covariance matrix can be used to define a portfolio's "risk" $w^\intercal \Sigma w$, which is precisely the variance in the returns it generates, assuming the underlying model is accurate. Denote the all-ones vector by $\mathbf{1}$, and for any pair of vectors u, v let $\langle u, v \rangle$ denote the standard inner product between u and v. The goal of the Markowitz formulation of PO is to find the optimal portfolio (i.e., vector of weights w) that either:

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- maximizes the expected return subject to a fixed risk parameter σ_0^2

$$\begin{aligned} &\max_w \langle w, r\rangle \\ \text{s.t.} \quad &w^\intercal \Sigma w = \sigma_0^2 \\ &\langle \mathbf{1}, w\rangle = 1 \end{aligned} \tag{8.1}$$

- minimizes risk subject to a fixed return parameter r_0

$$\begin{aligned} &\min_w w^\intercal \Sigma w \\ \text{s.t.} \quad &\langle w, r\rangle = r_0 \\ &\langle \mathbf{1}, w\rangle = 1 \end{aligned} \tag{8.2}$$

- maximizes return and minimizes risk with a tradeoff determined by a parameter known as the "risk aversion parameter" λ:

$$\begin{aligned} &\max_w \langle w, r\rangle - \lambda w^\intercal \Sigma w \\ \text{s.t.} \quad &\langle \mathbf{1}, w\rangle = 1 \end{aligned} \tag{8.3}$$

  or the alternative for the square root of risk (standard deviation rather than variance)

$$\begin{aligned} &\max_w \langle w, r\rangle - q\sqrt{w^\intercal \Sigma w} \\ \text{s.t.} \quad &\langle \mathbf{1}, w\rangle = 1, \end{aligned} \tag{8.4}$$

  where q plays the same role as λ.

Typically, it is satisfactory to find a vector that optimizes the objective function up to additive error ϵ, for some prespecified value of ϵ.

When solving the above Markowitz model formulations of PO, the absence of inequality constraints leads to simpler optimization problems that can be solved with efficient classical approaches. For example, the optimization problem in Eq. (8.2) is a simple quadratic program without complicated constraints, for which one can derive a closed-form expression for w using Lagrange multipliers [760]. More general PO problems that include practically relevant constraints (such as the simple "long-only" constraint $w_i \geq 0$, which prohibits "short" positions in which w_i can be less than zero) cannot generically be solved analytically, and one needs to employ more sophisticated numerical solvers. Real-world PO problems include a number of possible constraints (see [781] for a discussion), including, but not limited to:

- Long only—$w_j \geq 0$ for all j.
- Investment bands—$w_j \in [w_j^{\min}, w_j^{\max}]$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- Turnover constraints—$|\Delta w_j| \leq U_j$ for some fixed fraction U_j, where Δw_j represents the change in holdings of asset w_j from one portfolio to the next.
- Cardinality constraints—minimum, maximum, or exact number of nonzero assets in the portfolio.
- Sector constraints—specified minimum and/or maximum allocations to groups of assets (e.g., the energy or healthcare sectors).
- Transaction costs—typically represented as a function of $|\Delta w_j|$, and often added as a term in the objective function rather than as a constraint.
- Market impact—the effect on the price of an asset that a market participant has when buying or selling the asset. Related to liquidity, market impact can be seen as a type of transaction cost that arises when a transaction causes the price of the asset to move.

The PO can also be formulated in an "online" manner, where, for example, asset performance data arrives one day at a time, and one has the opportunity to update the portfolio at the end of each day [686].

As is often the case with optimization problems, the problem formulation *strongly* affects the solution strategy and the problem "hardness." If the PO problem is unconstrained and continuous (i.e., each w_i is a real number), then the problem is relatively easy. If convex inequality constraints, such as the long-only or turnover constraints, are imposed, then the problem is harder but can still be tackled by relatively efficient methods for convex optimization. By contrast, if one discretizes the problem (so that w now represents an integer number of asset shares or lots being traded), or if one applies some of the constraints above (such as integer-valued constraints like cardinality), then the problem becomes nonconvex and considerably harder to solve. In general, with discrete constraints, the problem can be formulated as an instance of an integer program (IP) (if all variables are discrete) or a mixed-integer program (MIP) (if some variables are discrete and others are continuous), which are NP-hard and therefore intractable to solve in polynomial-time (in n) under widely believed assumptions. Alternatively, given the IP formulation of the problem as a starting point, one can encode the integer variables in a binary representation, thereby allowing the problem to be formulated as a quadratic unconstrained binary optimization (QUBO) instance [881]. These formulations allow quantum algorithms for combinatorial optimization to be employed; for example, the MIP formulation can be solved with a branch-and-bound approach [247], and the QUBO formulation can be solved via Grover-type methods, or heuristically through (NISQ-friendly) quantum annealing approaches (e.g., [790]).

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost/complexity

An early approach to solving this optimization problem using a quantum algorithm was presented in [865], in which the Markowitz problem is written as minimizing risk with fixed return (Eq. (8.2)), and without other complicated constraints. This simple optimization problem boils down to an equality constrained convex program; it can be solved by introducing Lagrange multipliers and solving a linear system (represented by a matrix G) involving the input data r and Σ [865]. The approach of [865] is to use a quantum linear system solver (QLSS) and prepare the quantum state $|w\rangle$ whose amplitudes are proportional to the optimal weights w_i. The complexity to do so to error ϵ is $\widetilde{O}(\kappa\zeta \log(1/\epsilon))$, where κ is the condition number of the matrix G being inverted and $\zeta = \|G\|_F/\|G\|$ is the ratio of its Frobenius norm to its spectral norm. The $\widetilde{O}$ suppresses logarithmic factors, including a factor coming from applying unitaries that block-encode the matrix G in polylog(n) depth, essentially equivalent to the assumption that log-depth quantum random access memory (QRAM) is available. It is a priori unclear what the value of κ and ζ would be for actual PO instances and whether they depend on n, but the explicit logarithmic dependence of this complexity on n is appealing. However, a drawback of this approach is that it produces the quantum state $|w\rangle$ rather than an estimate for the optimal portfolio w. Learning the n entries of w to precision ϵ in 2-norm incurs multiplicative overhead of $\widetilde{O}(n/\epsilon)$ using quantum pure state tomography [49] for total time complexity $\widetilde{O}(n\kappa\zeta/\epsilon)$.[1]

When convex linear inequality constraints, such as long-only or turnover constraints, are included, the above approach will not work. However, a more sophisticated method can be applied, which first maps the PO instance to a convex program (specifically, a second-order cone program (SOCP)) and then makes use of interior point methods to solve the program. These interior point methods can be quantized, forming quantum interior point methods (QIPMs) [612, 68]. The QIPM is an iterative method, where each iteration involves solving a linear equation with a QLSS and classically reading out the solution with tomography. Thus, the procedure within each iteration is similar to the procedure above for solving the unconstrained PO problem, but the linear system to be solved is different (and changes with each iteration). A preliminary study of the effectiveness of this approach for PO was given by [611], followed by a

[1] Reference [865] suggests several possible nonstandard problems that can be solved with $|w\rangle$ without actually learning the entries of w, such as sampling values of i with large $|w_i|$, and estimating overlaps $\langle \tilde{w}, w\rangle$ with hypothesized portfolios $\tilde{w}$. In general, inner products $\langle u, w\rangle$ of arbitrary normalized vectors u with w can be learned to precision ϵ using overlap estimation [637] (an application of amplitude estimation), incurring multiplicative overhead of $O(1/\epsilon)$, but no explicit linear-in-n dependence. However, the practical utility of such tasks within the existing workflows of financial institutions is unclear.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

more extensive study in [328]. The QIPM produces an ϵ-optimal classical estimate for w, and has time complexity $\widetilde{O}(n^{1.5}\zeta\kappa\xi^{-1}\log(1/\epsilon))$, where κ and ζ are the maximum condition number and Frobenius-to-spectral-norm ratio for the matrices that must be inverted over the course of the algorithm, respectively, and ξ is the precision to which tomography must be performed. Note that in principle ξ can stay constant even as the overall precision estimate $\epsilon \to 0$ [328].

With the addition of discrete constraints, PO is instead formulated as a nonconvex MIP. MIPs are typically solved with a branch-and-bound approach (for a summary in a financial context, see, e.g., [311, Chapter 11]). Key to this approach is the ability to solve convex relaxations of the MIP (where the discrete constraints are dropped) in poly(n) time (perhaps via classical or quantum interior point methods for SOCPs, as above). To impose the discrete constraints, a tree is constructed and explored, where generating the children of a given node in the tree requires solving one of these relaxations. Thus, the number of convex relaxations that must be solved is proportional to the tree size T, which is generally exponentially large in n. Reference [247] (extending prior work of [776]) showed that a quantum algorithm can produce the same output while exploring quadratically fewer nodes, solving roughly $\widetilde{O}(\sqrt{T})$ convex relaxations (but doing so coherently, which could introduce overheads), for a total complexity of $\widetilde{O}(\sqrt{T}) \cdot \text{poly}(n)$. The value of T is instance dependent and requires empirical estimation: a preliminary numerical analysis of the value of T for a certain ensemble of PO instances up to $n = 56$ found that $T \sim 2^{0.14n}$ to $2^{0.20n}$ [247].

The algorithm for online PO of [686], which leverages the multiplicative weights update method, has time complexity scaling as $\widetilde{O}(\sqrt{n})$, a potentially quadratic speedup compared to the analogous classical algorithm. However, the quantum algorithm has worse dependence on the number of time steps.

The assessment of the number of qubits used by these algorithms requires a nuanced discussion of loading classical data. A key feature of all of the approaches above is that they require (repeatedly) accessing the classical data representing the historical stock information (i.e., the returns r and the covariance matrix Σ) in the quantum algorithm. The size of this data is typically $O(n^2)$. Loading can be performed using block-encoding dense matrices of classical data and QRAM, which achieves $O(\log(n))$ depth (time), at the expense of $O(n^2)$ space. Here, several caveats are inherited from the QRAM primitive. Moreover, for practical values of n, this $O(n^2)$ space cost could be prohibitively large, although it is possible this space cost could manifest as a dedicated QRAM hardware element of the device, rather than as part of the main processor. If log-depth QRAM of sufficient size is not desired or not

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

available, the data could instead be loaded with only $O(\log(n))$ space and in $O(n^2)$ time, but this overhead in time would likely preclude the possibility of quantum speedup at least in the first two cases, where the formulation is convex and classical poly(n)-time algorithms exist.

Existing resource estimates

A detailed, end-to-end resource analysis of the PO problem using QIPMs was performed in [328]. The authors followed the approach of [611] and performed a careful accounting of all quantum resources, including constant prefactors. The authors found that one needs $800n^2$ logical qubits, a T-depth of

$$(2 \times 10^8)\kappa\zeta n^{1.5}\xi^{-2} \log_2(n) \log_2(\epsilon^{-1}) \log_2(\kappa\zeta n^{14/27}\xi^{-1}),$$

and a T-count of

$$(7 \times 10^{11})\kappa\zeta n^{3.5}\xi^{-2} \log_2(n) \log_2(\epsilon^{-1}) \log_2(\kappa\zeta\xi^{-1}),$$

where κ is the maximum condition number encountered in the algorithm, ζ is the maximum Frobenius-to-spectral-norm ratio, and ξ is the minimum tomographic precision required. The ξ^{-2} dependence can asymptotically be improved to ξ^{-1} at the expense of a more sophisticated protocol for tomography [49]. Furthermore, these estimates used explicit bounds on the complexity of the QLSS from [313] which have since been improved upon in [572, 327]. Using the updated bounds from [327] would immediately reduce the T-count and T-depth estimates by at least three orders of magnitude. Note also that this calculation incorporated optimized circuits for block-encoding dense matrices of classical data with $O(\log(n))$ T-depth but $O(n^2)$ T-count [296], leading to the large discrepancy between those two quantities. The authors performed numerical simulations of PO instances to determine the instance-specific quantities. Using numerically determined values for $\kappa\zeta$ and ξ, and using realistic values of $\epsilon = 10^{-7}$ and $n = 100$, these resource counts imply that one would need 8×10^6 logical qubits, 2×10^{24} T-depth, and 8×10^{29} T-count. These logical estimates for the number of non-Clifford gates could in principle be turned into estimates for the number of physical qubits and runtime on actual hardware, using the methods discussed in the section on fault-tolerant quantum computation. However, the authors of [328] did not do so, in part because the logical costs were sufficiently high that the qualitative conclusion about the practicality of the algorithm was already clear.

Caveats

The quantum algorithms for PO discussed above inherit many of the caveats of their underlying primitives, namely, QLSS, tomography, and classical data

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

loading. One salient caveat is that the QLSS-based approaches depend on a number of instance-specific parameters κ, ζ, ξ, which are difficult to predict without running numerical simulations. The asymptotic speedup is subject to assumptions about the scaling of these parameters. Additionally, for a speedup to be possible, log-depth QRAM must be available on large datasets, which, while theoretically possible, presents practical challenges.

The branch-and-bound approach does *not* require log-depth QRAM to achieve its nearly quadratic speedup since the runtime will be dominated by the exponential tree-size factor (although it would help to have fast QRAM to reduce by poly(n) factors the time needed to solve the convex relaxations at each step). However, a caveat to that approach is that to obtain the quadratic speedup, the convex relaxations of the MIP (which would be SOCPs), would need to be solved coherently. In principle, this is always possible, but it would likely require a substantial amount of coherent classical arithmetic and additional poly(n) overheads in time and space.

Comparable classical complexity and challenging instance sizes

Convex formulations of the PO problem are typically solved classically via mapping to SOCPs. Optimized software packages can solve these SOCPs efficiently, and many are based on interior point methods. These interior point methods have theoretical runtime complexity of roughly $\widetilde{O}(n^{\omega+0.5}\log(1/\epsilon))$, where $\omega \approx 2.373$ is the matrix multiplication exponent, although for practical instance sizes, the effective value of ω is typically closer to 3. Note that the example PO problem with 100 assets solved in [328] and described above can typically be solved within seconds on a laptop using traditional classical methods. Problem sizes found in the financial services industry can include as many as tens of thousands of assets.

In the IP or MIP formulation of PO, classical solutions will have complexity exponential in n. As a point of reference, the numerical experiments reported in [247] classically solved hundreds of PO instances up to size $n = 56$ (and likely could have gone significantly higher).

Speedup

Recall that the QIPMs used to solve the SOCP for constrained PO are virtually identical to their classical counterpart; they differ by their use of a quantum subroutine to solve linear systems. Thus, any speedup obtained by the quantum approach to solving the SOCP will necessarily come from speedups from the QLSS plus tomography approach to solving a linear system. The approach for unconstrained PO was also based on the same primitives. The performance of the quantum method is often compared against classical Gaussian elimination.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

However, since the quantum approach necessarily produces an approximate solver (due to tomography), another valid comparison to make is against approximate classical solvers, such as the conjugate gradient method [527] or the randomized Kaczmarz method [959]. In the case of the randomized Kaczmarz method, the classical complexity for solving an $L \times L$ linear system to precision ξ scales as $O(L\kappa^2\zeta^2 \log(\xi^{-1}))$ (where κ is the condition number and ζ the Frobenius-to-spectral-norm ratio) compared to $O(L^3)$ for Gaussian elimination (asymptotically $O(L^\omega)$). Thus, the quantum method provides the greatest speedup when $\kappa\zeta \propto L$ and $\xi = O(1)$, in which case the QIPM for constrained PO runtime scales as $\widetilde{O}(n^{2.5})$, whereas the classical runtime scales as $\widetilde{O}(n^{3.5})$, where n is the number of stocks in the portfolio (see [328, Table XI] for a more complete discussion). For unconstrained PO, which only requires solving one linear system, the comparison would be $\widetilde{O}(n^2)$ vs. $\widetilde{O}(n^3)$. In either case, the speedup is subquadratic. Moreover, the numerical simulations in [328] were not consistent with these optimistic assumptions on $\kappa\zeta$ and ξ, suggesting, rather, that the QIPM would have minimal if any speedup over classical IPMs, albeit based on small instance sizes up to $n = 120$.

The speedup for the branch-and-bound approach to the MIP formulation of PO is quadratic (up to log factors), although, as mentioned, in contrast to the convex formulations, both the quantum and classical algorithms generally have runtime exponential in n.

NISQ implementations

Several alternative approaches to PO using quantum solutions have been proposed.

- Hybrid-HHL [1062]. This work generalizes the algorithm of [865], described above, by employing midcircuit measurements and conditional logic to obtain a NISQ version of the QLSS that readily solves the PO problem.
- Variational approaches based on the quantum approximate optimization algorithm (QAOA) [184, 524, 85] . These approaches typically use the quadratic objective function from Eq. (8.3), but instead consider $w_i \in \{0, 1\}$ as binary variables indicating whether or not an asset is part of the portfolio (a substantial deviation from the normal formulation). Constraints are dealt with by adding penalties to the objective function. Alternatively, constraints can be enforced by choosing clever versions of the ansatz [802] or by making measurements to project into the feasible space [524].
- Quantum annealing approaches: [881, 825, 824, 452, 790]. As in the previous case, these approaches require the problem to be formulated as a binary

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

optimization problem. However, in this case, they typically start with the IP formulation and encode integers in binary through one of several possible encodings [881] (thus, the number of binary variables will be greater than n). Constraints in the PO problem can also be included in the objective function using a variety of tricks, resulting in the desired QUBO, which can then be solved using a quantum annealer.

Outlook

The QIPM approach (and QLSS-based techniques more generally) for continuous formulations of PO have the potential to offer polynomial (but subquadratic) speedups for the PO problem. However, these speedups are subject to conjectures about the scaling of certain instance-specific parameters and preliminary empirical estimates are not suggestive of a maximal speedup. In any regard, the resource estimates of [328] illustrate that the non-Clifford resources required to implement the QIPM for this use case are prohibitive, even at problem sizes that are trivial to solve with classical computers. An asymptotic quantum advantage for this problem could exist for sufficiently large sets of assets, but without drastic improvements to the quantum algorithm and the underlying primitives (e.g., QRAM, QLSS), it is unlikely this approach will be fruitful. Even if such improvements are made, the algorithm only provides a polynomial speedup that is subquadratic, at best, greatly limiting the upside potential of this approach.

The branch-and-bound approach for discrete formulations has the possibility of a larger quadratic speedup, but, as has been observed (see, e.g., [223, 79]) in the context of Grover-like quadratic speedups in combinatorial optimization, it is unclear whether the quadratic speedup is sufficient to overcome the inherently slower quantum clock speeds and overheads due to fault-tolerant quantum computation for practical instance sizes.

8.2 Monte Carlo methods: Option pricing

Overview

Many financial instruments require an estimate of the average of some function of a stochastic variable within a window of time. To compute this average, one can use Monte Carlo methods to perform many simulations of the stochastic process over the time window, evaluate the function (which can potentially depend on the path taken by the stochastic variable during the entire window), and numerically estimate the average. While the setup and details of the problems may vary from one use case to another, the underlying methods are often

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

quite similar. As an archetypal example of this problem, we will focus on the problem of pricing derivatives, such as options, but we remark that many of these results can be carried over to other use cases, such as computing the Greeks, credit valuation adjustments, and value at risk.

Derivatives are financial instruments that, roughly speaking, allow the parties involved to benefit when an asset (such as a stock) increases or decreases in value, but without having to hold the asset itself. One type of derivative—called an "option"—is a contract that permits the holder to either purchase ("call option") or sell ("put option") an underlying asset at a fixed, predetermined price (the "strike price") at or prior to some predetermined time in the future ("the exercise window"). The seller of the option is obligated to either sell or buy the asset, should the holder choose to exercise the option.

How, then, should one decide on a price for the option (i.e., the amount the holder must pay for the contract, not the strike price)? The well-known Black–Scholes (or Black–Scholes–Merton) model provides one approach to pricing options, making a few assumptions about the underlying assets and the rules of the contract. More complicated options can be considered that include, for example, multiple assets in the contract (e.g., basket options), multiple possible exercise windows (e.g., Bermudan or American options), etc.

Typically, options are priced by running Monte Carlo sampling on the value of the underlying asset(s) and determining the expected profit or loss from a given position, which can be translated into a price that the purchaser must pay. Options with a larger potential downside for the seller should cost a larger amount to purchase. For more information on options and Monte Carlo methods in the context of computational finance, see [554, 435].

Actual end-to-end problem(s) solved

The task is to price an option based on an underlying asset. The price of the asset is a random variable X that follows a known (or assumed) stochastic process that models the market for the underlying asset. The option has a known payoff function $f(X)$ (e.g., the difference between the price of the asset at each time step minus the strike price over the trajectory, or zero, whichever is larger). For options that depend on more than one underlying asset or on asset prices at multiple distinct points in time, the random variable X would represent a vector of data containing all information needed to compute the payoff. Given these inputs, the end-to-end problem is to compute an estimate of the expected payoff $\mathbb{E}_X(f(X))$ that lies within a certain error tolerance ϵ with high probability. This quantity is then used to determine the fair price of the option, which we take to be the expected value of the derivative at the contract's expiration date, discounted to the pricing date.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Using the assumed stochastic model for the price of the asset, one can develop a stochastic differential equation for the average payoff of the option. In limited cases, one can compute the average payoff analytically, as in the case of the famous Black–Scholes formula for the price of European call options, for which the 1997 Nobel Prize in Economics was awarded. The Black–Scholes differential equation for the price of an asset at time t can be derived by assuming the price of the underlying stock follows a geometric Brownian motion

$$\mathrm{d}X_t = X_t \alpha \mathrm{d}t + X_t \sigma \mathrm{d}W_t \,,$$

where X_t is the price of the underlying asset at time t, α is a parameter known as the "drift" of the asset, σ is the volatility (the standard deviation of the underlying returns), and $\mathrm{d}W_t$ is an increment of an accompanying Brownian motion W_t. Using Itô's lemma, one can derive a differential equation for the price of the option at time t and, in limited cases (with several assumptions), one can solve the differential equation analytically. In practice, however, different types of contract have more complex definitions and fewer assumptions and, as a consequence, the differential equation cannot be solved analytically. Quantum approaches to numerically solving the stochastic differential equation have been proposed, including finite difference methods [767], Hamiltonian simulation [441], and quantum random walks [692]. For more detail on quantum approaches to solving differential equations, see Chapter 7 on solving differential equations. In many real-world derivative pricing use cases, the underlying differential equation becomes intractable. Thus, the most common classical method of computing the average payoff of an option is not through solving the stochastic differential equation, but rather through Monte Carlo sampling the random process X directly. To do so, one generates a large number of price trajectories over the chosen time range, and the average payoff is computed numerically. In what follows, we will focus on quantum approaches to Monte Carlo estimation—also known as quantum-accelerated[2] Monte Carlo methods—which was pioneered in [773] and subsequently applied to several problems in finance (e.g., [867, 1049, 949, 596, 950, 491]). However, we remark that other approaches to solving this problem that do not make use of Monte Carlo methods have also been proposed (e.g., [868]), and that this is an area of active research.

To compute different quantities, such as value at risk or credit valuation adjustments, similar approaches are often employed: simulate the underlying stochastic evolution several times and estimate the desired quantity numeri-

[2] Not to be confused with quantum Monte Carlo methods, which are classical algorithms for simulating certain quantum systems.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

cally. The function to be computed may be quite different, but the approach is often the same.

Dominant resource cost/complexity

In [867, 949], the quantum speedup of Monte Carlo estimation from [773] is applied to solve the option pricing problem. We briefly explain the method and its dominant cost. First of all, this requires discretizing the set of values the random variable X can take, which we index by the label x. Let N denote the number of values and $n = \lceil \log_2(N) \rceil$ denote the number of qubits needed to hold the state $|x\rangle$. The first step is to load the probabilities for the future prices of the asset into the amplitudes of a quantum state, that is, the state

$$\sum_x \sqrt{p_x}|x\rangle\,,$$

where p_x is the probability that x is observed in the corresponding classical Monte Carlo simulation.

Second, a subroutine is employed that computes information about the payoff function into an ancilla register using coherent arithmetic. More precisely, the angle θ_x is computed (rounded to some finite number of bits of precision), where $\sin(\theta_x) = \sqrt{f(x)}$. (For simplicity, here we assume $0 \le f(x) \le 1$ for all x, but we revisit this point later.) This yields

$$\sum_x \sqrt{p_x}|x\rangle|\theta_x\rangle\,.$$

Third, the amplitude $\sqrt{f(x)}$ is loaded into the amplitude of an ancilla register by applying the map $|\theta\rangle|0\rangle \mapsto |\theta\rangle(\sin(\theta)|0\rangle + \cos(\theta)|1\rangle)$. This gives

$$\left(\sum_x \sqrt{p_x f(x)}|x\rangle|\theta_x\rangle\right)|0\rangle + \left(\sum_x \sqrt{p_x(1-f(x))}|x\rangle|\theta_x\rangle\right)|1\rangle\,.$$

The probability of measuring the final ancilla in $|0\rangle$ is precisely $\mathbb{E}_X(f(X))$. Thus, the final step is to apply quantum amplitude estimation, which requires $O(\epsilon^{-1})$ calls to the unitary that produces the state above to obtain an estimate to error ϵ.

If $0 \le f(x) \le 1$ does not hold, the above approach needs to be modified, for example, by shifting and rescaling f over a sequence of intervals of increasing length, as discussed in [773, 867]. Roughly speaking, to make sure that $f(x)$ falls within the interval $[0, 1]$, at least for a large fraction of the randomly chosen values of x, we should expect the function f will need to be scaled down by a factor on the order of the standard deviation $\sigma = \sqrt{\mathbb{E}_X(f(X)^2) - (\mathbb{E}_X f(x))^2}$. Thus, to achieve error ϵ, quantum amplitude estimation must be performed to precision ϵ/σ instead of ϵ.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

There are three components to the algorithm that each contribute to the resource cost:

- Loading the distribution with amplitudes $\sqrt{p_x}$. The gate complexity of this step is roughly the same as the time complexity of classically drawing a Monte Carlo sample, although for certain distributions it could be faster (e.g., a quadratic quantum speedup can be obtained if p_x is the stationary distribution of a Markov process [974]). Alternatively, if a functional form for p_x is known, the methods of [754] could be used to approximately prepare the state—note that the Grover–Rudolph approach to state preparation [463] is incompatible with a quantum speedup in the context of Monte Carlo estimation [521]. Finally, [949] proposes using a quantum generative adversarial network (qGAN), a variational quantum algorithm, which could reduce the resources but requires a training phase.
- Coherent arithmetic to compute the rotation angle θ_x. This depends on the complexity of the function f, but can generally be accomplished in comparable gate complexity as classical arithmetic, that is, poly(n). In [948], it was shown how the payoff can instead be put directly into the amplitude, without ever computing θ_x, using quantum signal processing methods [754].
- Quantum amplitude estimation to precision ϵ/σ, which requires $O(\sigma/\epsilon)$ repetitions of the above two costs to achieve an ϵ-estimate on the quantity $\mathbb{E}_X f(X)$.

Overall, from [642, Theorem 1.1] the complexity is

$$\frac{\sigma}{\epsilon} \cdot \text{poly}(n)\,, \tag{8.5}$$

with the poly(n) factor generally on the same order as the time required to draw and process a single classical Monte Carlo sample. This complexity does not require one to have an upper bound on σ, and it improves over the original work of [773] (which in turn is based on the algorithm of [515] for the uniform distribution) and follow-up work [490] by removing additional $\log(\sigma/\varepsilon)$ factors originating from the need to rescale the (potentially unbounded) random variable, to account for the contribution of its tails. In fact, the method can also work even for random variables with infinite variance [156]. However, in practice, the more advanced techniques and analyses of [642, 156] may not be necessary, as the underlying assets are typically modeled with distributions such as Gaussians where the tails are well behaved and the variance of the relevant random variable is controlled.

The general approach to Monte Carlo estimation sketched above has been extended and optimized in various ways; see, for example, [768, 309, 522,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

361, 14]. For instance, in [361], the authors study a quantum algorithm for the optimal stopping problem by developing a quantum version of least-squares Monte Carlo. The algorithm finds a quadratic speedup over related classical methods, thereby demonstrating that American-style options—which are more complex than European-style options because they allow the holder to exercise the option and buy/sell the underlying asset at any point in the exercise window, rather than just the end—also maintain a quadratic speedup over classical Monte Carlo.

Existing resource estimates

Detailed resource estimations for benchmark option pricing problems (known as autocallable and target accrual redemption forward, or TARF) were studied in [246]. The authors studied real-world use cases and problem sizes that are simple enough to analyze, yet complex enough to capture desirable features (such as path dependence and multiple underlyings), making them relevant to current financial institutions. For a basket autocallable with 3 underlying assets, 5 payment days, and a knock-in put option with 20 barrier dates, the authors found that one would need about 8000 logical qubits, a T-depth of 5.4×10^7, and a T-count of about 1.2×10^{10}, using the most efficient methods they studied. For a TARF with 1 underlying and 26 payment dates, one needs about 1.2×10^4 logical qubits, a T-depth of about 8.2×10^7, and a T-count of about 9.8×10^9. A follow-up analysis [948] involving a quantum signal processing approach subsequently reduced these estimates to 4.7×10^3 logical qubits, 4.5×10^7 T-depth, and 2.4×10^9 T-count. For comparison, classical Monte Carlo methods are roughly estimated to require 1–10 seconds for 4×10^4 samples to achieve the same accuracy on these examples.

Similar analyses were performed in [950] for the computation of the Greeks, which are quantities that measure the sensitivity of a derivative to various parameters. To compute the Greeks of an option, one needs to compute the derivative of the payoff function with respect to, for example, the price of the underlying. To to do this on a quantum computer, one needs to be able to estimate both the expectation of the payoff function and have a way of computing gradients. The authors apply several quantum methods of computing gradients in order to calculate the Greeks, in addition to the quantum approaches to Monte Carlo methods used. Using a quantum gradient method to compute Greeks of an option, the authors estimate that one would need about 1.2×10^4 logical qubits and a T-depth of around 10^8.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Caveats

There are many types of options and derivatives that may not be accurately captured by these simple models. Some payoff functions are path dependent, and hence one cannot simply use the asset value at some fixed time to compute the cost, but rather the cost depends on the trajectory the random variable takes in each Monte Carlo sample.

Moreover, classical approaches to Monte Carlo sampling often allow for massive parallelization, as each simulation of the underlying asset can be done independently. By contrast, quantum algorithms for this problem require a *serial* approach, as the subroutines in the quantum algorithm must be run one after another without measurement and restart if the quadratic advantage is to be realized. When the slower clock speed found in quantum devices is also taken into account, the requirements for a quantum speedup over classical methods become more stringent, as much larger problem sizes are required to achieve practical advantage. For further reading, see [176, Section 2.3], for example.

It is worth noting that in certain cases the number of classical samples needed to achieve error ϵ can be reduced from the naive $O(\sigma^2/\epsilon^2)$, cutting into the quadratic quantum speedup. In particular, quasi–Monte Carlo methods, which sample possible trajectories of the underlying assets nonrandomly, can achieve a nearly quadratic speedup compared to traditional classical Monte Carlo methods, but gain an exponential dependence on the number of underlying assets ("curse of dimensionality") which limits their use; see [435, Chapter 5]. The number of samples can also potentially be reduced classically via multilevel Monte Carlo methods [426], although a quantum algorithm for multilevel Monte Carlo also exists [33]. In general, when and how these various methods work is delicate and must be evaluated on a case-by-case basis.

Comparable classical complexity and challenging instance sizes

Classical approaches to option pricing comprise some of the largest computational costs incurred by financial institutions. In the traditional approach to solving the option pricing problem, Monte Carlo sampling is required to simulate the evolution of the underlying asset over the time horizon of the option, and it can be slow to converge. In particular, denote the expectation value of $f(X)$ by $V := \mathbb{E}_X(f(X))$, and the variance of $f(X)$ by σ^2. Classical Monte Carlo methods compute an estimate $\hat{V}$ for V formed by averaging $f(X)$ for M independent samples of X. By Chebyshev's inequality,

$$\Pr(|V - \hat{V}| \geq \epsilon) \leq \frac{\sigma^2}{M\epsilon^2}.$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Thus, classically one needs $M \sim O(\sigma^2/\epsilon^2)$ samples to find an estimate $\hat{V}$ within a 99% confidence interval [773].

In typical industrial scenarios, options can be priced to sufficient operational precision after roughly a few seconds of runtime, sampling as many as tens of thousands of Monte Carlo trajectories.

Alternatively, a tensor network–based classical approach to option pricing was proposed by [602] that could lead to significant advantages over traditional classical methods in some cases.

Speedup

The classical algorithm requires $M = O(\sigma^2/\epsilon^2)$ samples whereas the quantum algorithm requires only $\widetilde{O}(\sqrt{M}) = \widetilde{O}(\sigma/\epsilon)$ samples. The gate cost of a sample is roughly the same classically and quantumly, and thus the speedup is (nearly) quadratic, inherited from the quadratic speedup of quantum amplitude estimation.

Outlook

In [246], the authors place an upper bound on the resources required for pricing options on quantum computers, and they provide a goalpost for quantum hardware development to be able to outperform classical Monte Carlo methods. In particular, the authors estimate that a quantum device would need to be able to execute about 10^7 layers of T gates per second. Moreover, the code distance for fault-tolerant implementation would need to be chosen large enough to support 10^{10} total error-free logical operations. These requirements translate to a logical clock rate of about 50 MHz that would be needed in order to compete with current classical Monte Carlo methods. This clock speed is orders of magnitude faster than what is foreseeably possible given the current status of physical hardware and currently known methods for performing logical gates in the surface code.

While the resource requirements for pricing of derivatives are quite stringent, this is nevertheless an area of active research. For example, a new "analog" quantum representation of stochastic processes was developed in [177] that can compute ϵ-accurate estimates of time averages (over T time steps) of certain functions of stochastic processes in time $\text{polylog}(T) \cdot \epsilon^{-c}$, where $3/2 < c < 2$, an exponential speedup over classical methods in the parameter T. The analog nature of their method leads to additional caveats, and finding concrete applications of this method remains an interesting open question.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

9

Machine learning with classical data

There has been significant recent interest in exploring the interplay between quantum computing and machine learning. Quantum resources and quantum algorithms have been studied in all major parts of the traditional machine learning pipeline: (i) the dataset, (ii) data processing and analysis, (iii) the machine learning model leading to a hypothesis family, and (iv) the learning algorithm (see [151, 242, 293] for reviews). In this chapter, we predominantly focus on quantum approaches for the latter three categories—that is, here we mostly consider quantum algorithms applied to classical data. These approaches include algorithms hinging on the quantum linear system solver (or quantum linear algebra more generally) as the source for possible quantum speedup over classical learning algorithms. These also include *quantum neural networks* (using the framework of variational quantum algorithms) and *quantum kernels*, where the classical machine learning model is replaced with a quantum model. Additionally, in this chapter, we discuss quantum algorithms that aim to speed up data analysis tasks, namely, *tensor principal component analysis (TPCA)* and *topological data analysis*.

Quantum machine learning is an active area of research. As such, we expect the conclusions made in this chapter to evolve over time, as new results are discovered. At present, our evaluation suggests that few of the considered quantum machine learning algorithms show any promise of quantum advantage in the immediate future. This conclusion stems from a number of factors, including issues of loading classical data into the quantum device and extracting classical data via tomography, and the success of classical "dequantized" algorithms [976]. More specialized tasks such as tensor PCA and topological data analysis may provide larger polynomial speedups (i.e., better than quadratic) in some regimes, but their application scope is less broad. Finally, other techniques such as quantum neural networks and quantum kernel meth-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ods contain heuristic elements which make it challenging to perform concrete analytic end-to-end resource estimates [915].

The authors are grateful to Ewin Tang for reviewing Section 9.1, to Eric Anschuetz for reviewing Section 9.2, to Matthew Hastings and Robin Kothari for reviewing Section 9.3, to Vedran Dunjko for reviewing Sections 9.4 and 9.5, and to Marco Cerezo for reviewing Section 9.5.

9.1 Quantum machine learning via quantum linear algebra

Overview

Linear algebra in high-dimensional spaces with tensor product structure is the workhorse of quantum computation as well as of much of machine learning (ML). It is therefore unsurprising that efforts have been made to find quantum algorithms for various learning tasks, including but not restricted to cluster-finding [707], principal component analysis [708], least-squares fitting [917, 609], recommendation systems [608], binary classification [866], and Gaussian process regression [1089]. One of the main computational bottlenecks in all of these tasks is the manipulation of large matrices. Significant speedup for this class of problems has been argued for via quantum linear algebra, as exemplified by the quantum linear system solver (QLSS). The main question marks for viability are (i) can quantum linear algebra be fully dequantized [977] for ML tasks, (ii) can the classical training data be loaded efficiently into a quantum random access memory (QRAM), and (iii) do the quantum ML algorithms that avoid the above-mentioned pitfalls address relevant machine learning problems? Our current understanding suggests that significant quantum advantage would require an exceptional confluence of (i)–(iii) that has not yet been found in the specific applications analyzed to date, though modest speedups are plausible.

ML applications

The structure of this section differs from other sections in Part I, due to the disparate nature of many of the quantum machine learning proposals and the fact that they are often heuristic. Rather than cover every proposal, we explore three specific applications. Each example explains which end-to-end problem is being solved and roughly how the proposed quantum algorithm solves that problem, arriving at its dominant complexity. In each case, the quantum algorithm assumes access to fast coherent data access (log-depth QRAM) and leverages quantum primitives for solving linear systems (and linear algebra

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

more generally). Under certain conditions, these primitives can be exponentially faster than classical methods that manipulate all the entries of vectors in the exponentially large vector space. However, for these examples, it is crucial to carefully define the end-to-end problem, as exponential advantages can be lost at the readout step, where the answer to a machine learning question must be retrieved from the quantum state encoding the solution to the linear algebra problem. In the three examples below, this is accomplished with some form of amplitude or overlap estimation, a primitive that brings a multiplicative $O(1/\epsilon)$ factor into the overall complexity when seeking precision ϵ. This $O(1/\epsilon)$ readout cost could be avoided if it were the case that $O(1)$ *samples* from the output state prepared by quantum linear algebra were sufficient for solving the end-to-end problem—situations where this may arise include training large neural networks by solving nonlinear differential equations [701] and determining an optimal set of random features in kernel-based supervised learning [1064]—but we do not cover these examples in detail here.

Furthermore, even if these quantum algorithms are exponentially faster than classical algorithms that manipulate the full state vector, in some cases this speedup has been "dequantized" via classical algorithms that merely sample from the entries of the vector. Specifically, for some end-to-end problems, there exist classical "quantum-inspired" algorithms [977, 271, 924] that solve the problem in time only polynomially slower than the quantum algorithm assuming an analogous data-input model. Namely, the assumption that the quantum algorithm has fast QRAM access to the classical data is analogous to the assumption that the classical algorithm has fast "sample-and-query" (SQ) access to the data—SQ access allows the classical algorithm to sample an entry from the database with probability proportional to its value squared, or to compute the value of any specific entry of the database. The reason that it is fair to compare quantum algorithms relying on QRAM access with classical algorithms relying on SQ access is that both utilize a certain tree-like data structure to enable fast implementation at the circuit level. A large, one-time cost (scaling linearly in the total size of the classical dataset) may be required to "load" the data structure with the classical data, but the data structure is dynamic in the sense that if a single entry in the database is added or changed, updating the data structure has low cost (scaling logarithmically in the total size of the classical dataset). Once the data structure has been set up, one can implement QRAM access (resp. SQ access) using a quantum (resp. classical) circuit with depth only logarithmic in the size of the database. We will not cover the particulars of the quantum-inspired algorithms in more detail, but we note that most of the machine learning tasks based on linear algebra for which quantum

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

algorithms have been proposed have also been dequantized in some capacity [271].

However, it is worth emphasizing that in some cases it remains possible that there could be an exponential quantum advantage if the quantum algorithm is able to exploit additional structure in the matrices involved, such as sparsity, that the classical algorithm cannot. The three examples below roughly illustrate the spectrum of possibilities: some tasks are fully dequantized, whereas others, to the best of our current knowledge, could still support exponential advantages if certain conditions are met.

Example 1: Gaussian process regression

Actual end-to-end problem: Gaussian process regression (GPR) is a non-parametric, Bayesian method for regression. GPR is closely related to kernel methods [594], as well as to other regression models, including linear regression [858]. Our presentation of the problem follows that of [858, Chapter 2] and [1088]. Given training data $\{x_j, y_j\}_{j=1}^M$, with inputs $x_j \in \mathbb{R}^N$ and noisy outputs $y_j \in \mathbb{R}$, the goal is to model the underlying function $f(x)$ generating the output y

$$y = f(x) + \epsilon_{\text{noise}},$$

where ϵ_{noise} is drawn from i.i.d. Gaussian noise with variance σ^2. Modeling $f(x)$ as a Gaussian process means that for inputs $\{x_j\}_{j=1}^M$, the outputs $\{f(x_j)\}_{j=1}^M$ are treated as random variables with a joint multivariate Gaussian distribution, in such a way that any subset of these values are jointly normally distributed in a manner consistent with the global distribution. While this multivariate Gaussian distribution governing $\{f(x_j)\}_{j=1}^M$ will generally be correlated for different j, the additional additive error ϵ_{noise} on our observations y_j is independent from the choice of $f(x_j)$ and uncorrelated from point to point. The Gaussian process is specified by the distribution $\mathcal{N}(m, K)$ where m is the length-M vector obtained by evaluating a "mean function" $m(x)$ at the points $\{x_j\}_{j=1}^M$, and K is an $M \times M$ covariance kernel matrix obtained by evaluating a covariance kernel function $k(x, x')$ at $x, x' \in \{x_j\}_{j=1}^M$—$\mathcal{N}$ then denotes the multivariate Gaussian distribution with the corresponding mean and covariance. The functional form of the mean and covariance kernel are specified by the user and determine the properties of the Gaussian process, such as its smoothness.[1] These functions typically contain a small number of hyperparameters which can be optimized using the training data. A commonly used covariance kernel function is the squared exponential covariance function $k(x, x') = \exp\left(-\frac{1}{2\ell^2}\|x - x'\|^2\right)$ where

[1] This can be visualized by sampling a function from the distribution, which means sampling a value of $f(x_j)$ from the distribution for each x_j, and plotting the values of $f(x_j)$ as a curve.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ℓ is a hyperparameter controlling the length scale of the Gaussian process, and $\|\cdot\|$ denotes the standard Euclidean norm in the case of vector arguments and the spectral norm in the case of matrix arguments.

Given choices for $m(x)$ and $k(x, x')$ and the observed data $\{x_j, y_j\}_{j=1}^M$, our task is to predict the value $f(x_*)$ of a new test point x_*. Because the Gaussian process assumes that all $M + 1$ values $\{f(x_1), \ldots, f(x_M), f(x_*)\}$ have a jointly Gaussian distribution, it is possible to condition upon the observed data to obtain the distribution for $f(x_*)$ which is $p(f_*|x_*, \{x_j, y_j\}) \sim \mathcal{N}(\bar{f}_*, \mathbb{V}[f_*])$. Our goal is to compute $\bar{f}_*$, the mean (linear predictor) of the distribution for $f(x_*)$, as well as the variance $\mathbb{V}[f_*]$, which gives uncertainty on the prediction. Computing the underlying multivariate Gaussian distribution can be bypassed by exploiting the closure of Gaussians under linear operations, in particular, conditioning. This re-expresses the problem as linear algebra with the kernel matrix. Assuming the common choice of $m(x) = 0$ and defining the length-M vector $k_* \in \mathbb{R}^M$ to have its j-th entry given by $k(x_*, x_j)$, we obtain

$$\bar{f}_* = k_*^\intercal [K + \sigma^2 I]^{-1} y$$
$$\mathbb{V}[f_*] = k(x_*, x_*) - k_*^\intercal [K + \sigma^2 I]^{-1} k_*$$

which characterize the prediction for the test point. The advantages of GPR are a small number of hyperparameters, model interpretability, and that it naturally returns uncertainty estimates for the predictions. Its main disadvantage is the computational cost.

Dominant resource cost/complexity: In classical implementations, the cost is dominated by performing the inversion $[K+\sigma^2 I]^{-1}$, typically via a Cholesky decomposition, resulting in a complexity of $O(M^3)$ (see [858, Chapter 8] and [698] for approximations used to reduce the classical cost). In [1089], a quantum algorithm was proposed that leverages the QLSS to perform this inversion more efficiently. The quantum computer uses the classical data to infer the linear predictor and variance for a test point x_*, and this process must be repeated for the computation of each new test point output. We analyze the complexity of computing $\bar{f}_*$, with a simple extension for $\mathbb{V}[f_*]$. Given classically observed/precomputed values of y and k_*, the quantum algorithm uses state preparation from classical data (based on QRAM) to prepare quantum states representing $|y\rangle$ and $|k_*\rangle$,[2] each with a gate depth of $O(\log(M))$ (though using $O(M)$ gates overall). The algorithm also uses a block-encoding of classical data (also using QRAM) for $A := [K + \sigma^2 I]$, with a normalization factor

[2] For any vector v, the notation $|v\rangle$ denotes the normalized quantum state whose amplitudes in the computational basis are proportional to the entries of v, for example, $|y\rangle = \frac{1}{\|y\|} \sum_j y_j |j\rangle$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

of $\alpha = \|K + \sigma^2 I\|_F$, where $\|\cdot\|_F$ denotes the Frobenius norm.[3] The state-of-the-art QLSS has complexity $O(\alpha\kappa\|A\|^{-1}\log(1/\epsilon))$ calls to an α-normalized block-encoding of matrix A with condition number κ (see Eq. (18.2)). In this case, the minimum singular value of A is at least σ^2, so $\kappa/\|A\| \leq \sigma^{-2}$. The QLSS produces the normalized state $|A^{-1}y\rangle$, and a similar approach yields an estimate for the norm $\|A^{-1}y\|$ to relative error ϵ at cost $\widetilde{O}(\alpha\kappa\|A\|^{-1}\epsilon^{-1})$ [248, 327]. Given unitary circuits performing these tasks, we can estimate the quantity $\bar{f}_* = \langle k_*|A^{-1}y\rangle \cdot \|k^*\| \cdot \|A^{-1}y\|$ to precision ϵ using overlap estimation with gate depth upper bounded by

$$\widetilde{O}\left(\log(M) \cdot \|K + \sigma^2 I\|_F \sigma^{-2} \cdot \frac{\|k_*\|\left\|[K + \sigma^2 I]^{-1}y\right\|}{\epsilon}\right),$$

where the three factors come from state preparation (i.e., QRAM), QLSS, and overlap estimation, respectively. Using QRAM for state preparation as described above would use $O(M^2)$ ancilla qubits. Note that classical "quantum-inspired" methods for solving linear systems, based on SQ access, also have $\text{poly}(\|A\|_F, \kappa, \epsilon^{-1}, \log(M))$ complexity [271, 433, 924], and thus the quantum algorithm as stated above offers at most a polynomial speedup in the case of dense matrices.

On the other hand, [1089] considers the case where the vectors and kernels are sparse[4] and uses this to reduce the cost of the quantum algorithm and of QRAM. In this case, using block-encodings of sparse matrices, the factor $\|A\|_F$ in the complexity expression is replaced by a factor $s\|A\|_{\max}$, where s is the sparsity of the matrix A and $\|A\|_{\max}$ is the maximum magnitude of any entry of A—log-depth QRAM with $\Omega(M)$ ancilla qubits would still be necessary to implement the sparse access oracle to the sM arbitrary nonzero entries of A in depth $O(\log(M))$. The upshot is that in the sparse case, because the algorithm assumes the kernel is not low rank, this algorithm is not dequantized by SQ access [271] and may still offer an exponential speedup over quantum-inspired methods. However, we note that the assumption of sparsity in $[K + \sigma^2 I]$ may also enable the use of more efficient classical algorithms for computing the inverse (see Chapter 18 on QLSSs). Moreover, we must include the classical precomputation of evaluating the entries of this matrix. A related, and similarly efficient, quantum algorithm is proposed in [1088] for optimizing the

[3] It may be more efficient to load in the $\{x_j\}$ values and then coherently evaluate the kernel entries using quantum arithmetic. Some ideas in this direction are explored in [264]. One might also consider block-encoding K and $\sigma^2 I$ separately and combining them with linear combination of unitaries.

[4] For the squared exponential covariance function mentioned above, the kernel matrix will not be sparse, but [1089] notes several applications of GPR where sparsity is well justified.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

hyperparameters of the GPR kernel by maximizing the marginal likelihood of the observed data given the model.

Example 2: Support vector machines

Actual end-to-end problem: The task for the support vector machine (SVM) is to classify an N-dimensional vector x_* into one of two classes ($y_* = \pm 1$), given M labeled data points of the form $\{(x_j, y_j) : x_j \in \mathbb{R}^N, y_j = \pm 1\}_{j=1,\dots,M}$ used for training. The training phase solves a continuous optimization problem to find a maximum-margin hyperplane, described by normal vector $w \in \mathbb{R}^M$ and offset $b \in \mathbb{R}$, which separates the training data. That is, data points with $y_j = 1$ lie on one side of the plane, and data points with $y_j = -1$ lie on the other side. Once trained, the classification of x_* is inferred via the formula

$$y_* = \text{sign}(b + \langle w, x_* \rangle), \tag{9.1}$$

where $\langle \cdot, \cdot \rangle$ denotes the standard inner product between vectors.

In the "hard-margin" version of the problem where all training points must be classified correctly (assuming it is possible to do so, i.e., the data is linearly separable), the solution (w, b) is given by

$$\underset{(w,b)}{\text{argmin}} \, \|w\|^2, \qquad \text{subject to:} \qquad (\langle w, x_j \rangle + b) y_j \geq 1 \qquad \forall j. \tag{9.2}$$

In the "soft-margin" version of the problem, the hyperplane need not correctly classify all training points. The relation $(\langle w, x_j \rangle + b) y_j \geq 1$ is relaxed to $(\langle w, x_j \rangle + b) y_j \geq 1 - \xi_j$, with $\xi_j \geq 0$. Now, (w, b) are determined by

$$\underset{(w,b,\xi)}{\text{argmin}} \, \|w\|^2 + \gamma \|\xi\|_1, \quad \text{subject to:} \quad (\langle w, x_j \rangle + b) y_j \geq 1 - \xi_j \quad \forall j, \tag{9.3}$$

where $\|\cdot\|_1$ denotes the vector 1-norm, and γ is a user-specified hyperparameter related to how much to penalize points that lie within the margin. Both Eqs. (9.2) and (9.3) are convex programs, in particular, quadratic programs, which can also be rewritten as second-order cone programs [612]. Another feature of these formulations is that the solution vectors w and ξ are usually sparse; the j-th entry is only nonzero for values of j where x_j lies on or within the margin near the hyperplane—these x_j are called the "support vectors."

In [972], a "least-squares" version of the SVM problem was proposed, which has no inequality constraints:[5]

$$\underset{(w,b,\xi)}{\text{argmin}} \, \|w\|^2 + \frac{\gamma}{M} \|\xi\|^2, \quad \text{subject to:} \quad (\langle w, x_j \rangle + b) y_j = 1 - \xi_j \quad \forall j. \tag{9.4}$$

[5] Our definition of the least-squares SVM is equivalent to the normal presentation found in [972, 866]; however, we choose slightly different conventions for normalization of certain parameters, such as γ, with respect to M. The goal of our choices is to make the final complexity expression free of any explicit M dependence.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

This is an equality-constrained least-squares problem, which is simpler than a quadratic program and can be solved using Lagrange multipliers and inverting a linear system. Specifically, one introduces a vector $\beta \in \mathbb{R}^M$ and solves the $(M+1) \times (M+1)$ linear system $Au = v$, where

$$A = \begin{pmatrix} 0 & \mathbf{1}^\intercal / \sqrt{M} \\ \mathbf{1} / \sqrt{M} & K/M + \gamma^{-1} I \end{pmatrix}, \qquad u = \begin{pmatrix} b \\ \beta \end{pmatrix}, \qquad v = \frac{1}{\sqrt{M}} \begin{pmatrix} 0 \\ y \end{pmatrix}, \tag{9.5}$$

with K the kernel matrix for which $K_{ij} = \langle x_i, x_j \rangle$, $\mathbf{1}$ the all-ones vector, and I the identity matrix. The vector w is inferred from β via the formula $w = \sum_j \beta_j x_j / \sqrt{M}$.

However, unlike the first two formulations, the least-squares formulation does not generally have sparse solution vectors (w, b) (see [971]). Additionally, its solution can be qualitatively different, due to the fact that correctly classified data points can lead to negative ξ_j that apply penalties to the objective function through the appearance of $\|\xi\|^2$.

Dominant resource cost/complexity: The hard-margin and soft-margin formulations of SVM are quadratic programs, which can be mapped to second-order cone programs and solved with quantum interior point methods (QIPMs). This solution was proposed in [612], and, assuming access to log-depth QRAM it can find ϵ-accurate estimates for the solution (w, b) in time scaling as $\widetilde{O}(M^{0.5}(M+N)\kappa_{\text{IPM}}\zeta \log(1/\epsilon)/\xi')$, where κ_{IPM}, ζ, and ξ' are instance-specific parameters related to the QIPM. This compares to $O(M^{0.5}(M+N)^3 \log(1/\epsilon))$ for naively implemented classical interior point methods. In [612], numerical simulations on random SVM instances were performed to compute these instance-specific parameters, and the results were consistent with a small polynomial speedup. However, the resource estimate of [328] for a related problem suggests a practical advantage may be difficult to realize with this approach.

The least-squares formulation can be solved directly with the QLSS, as pursued in [866]. This can be compared to classically solving the linear system via Gaussian elimination, with cost $O(M^3)$. The QLSS requires the ability to prepare the state $|v\rangle$, which can be accomplished in $O(\log(M))$ depth through methods for preparation of states from classical data, although requiring $O(M)$ total gates and ancilla qubits. One also needs a block-encoding of the matrix A. One method is through block-encodings from classical data, which requires classical precomputation of the $O(M^2)$ entries of K (incurring classical cost $O(M^2 N)$) and producing a block-encoding with normalization factor $\alpha = \|A\|_F$ (Frobenius norm). Henceforth, we assume that $\|x_j\| \leq 1$ for all j, which can always be achieved by scaling down the training data (inducing a

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

scaling up of w and $\sqrt{\gamma}$ by an equal factor). This implies $\|K/M\|_F \leq 1$ and hence $\|A\|_F \leq \sqrt{2} + 1 + \sqrt{M}\gamma^{-1}$. A better block-encoding can be obtained by block-encoding K/M via the method for Gram matrices[6] and $\gamma^{-1}I$ via the trivial method, and then combining these with the rest of A via linear combination of block-encodings. This avoids the need to classically calculate the inner products $\langle x_i, x_j\rangle$, and has a better normalization $\alpha \leq \sqrt{2} + 1 + \gamma^{-1}$.

Given these constructions, the QLSS outputs the state $|u\rangle = (b|0\rangle + \sum_{j=1}^{M} \beta_j |j\rangle)/\sqrt{b^2 + \|\beta\|^2}$; the cost is $\widetilde{O}(\alpha\kappa_A/\|A\|)$ queries to the block-encoding of A, where κ_A is the condition number of A. We may assert that $\|A\| \geq 1$. This follows by noting that the lower-right block of A (as defined in Eq. (9.5)) is positive semidefinite, and that 1 is an eigenvalue of A when the lower-right block is set to zero. The condition number should be upper bounded by an M-independent function of γ due to the appearance of the regularizing $\gamma^{-1}I$.

Reading out all $M + 1$ entries of $|u\rangle$ via tomography would multiply the cost by $\Omega(M)$. However, in [866], it was observed that to classify a test point x_* via Eq. (9.1), one can use overlap estimation rather than classically learning the solution vector. In our notation and normalization, this can be carried out as follows. Let $|x_j\rangle = \sum_{i=1}^{M} x_{ji}|i\rangle/\|x_j\|$, with x_{ji} denoting the i-th entry of the vector x_j. Starting with $|u\rangle$, we prepare $|x_j\rangle$ into an ancilla register, using methods for controlled state preparation from classical data, forming

$$|\tilde{u}\rangle = \frac{b|0\rangle|0\rangle + \sum_{j=1}^{M} \beta_j |j\rangle\Big(\|x_j\||x_j\rangle + \sqrt{1 - \|x_j\|^2}|M+1\rangle\Big)}{\sqrt{b^2 + \|\beta\|^2}}.$$

One also creates a reference state $|\tilde{x}_*\rangle$ encoding x_*, defined as

$$|\tilde{x}_*\rangle = \frac{1}{\sqrt{2}}|0\rangle|0\rangle + \frac{1}{\sqrt{2M}} \sum_{j=1}^{M} |j\rangle\Big(\|x_*\||x_*\rangle + \sqrt{1 - \|x_*\|^2}|M+2\rangle\Big).$$

The right-hand side of Eq. (9.1) is then given by $\sqrt{2}\sqrt{b^2 + \|\beta\|^2}\langle\tilde{u}|\tilde{x}_*\rangle$. Thus, the overlap $\langle\tilde{u}|\tilde{x}_*\rangle$ must be estimated to precision $\epsilon = 1/\sqrt{2(b^2 + \|\beta\|^2)}$ in order to distinguish ± 1 and classify x_*. Additionally, the norm $\|u\| = \sqrt{b^2 + \|\beta\|^2}$ must be calculated; this can separately be done to relative error ϵ' at cost

[6] We sketch a possible instantiation of this method here. Define $|x_i\rangle = \|x_i\|^{-1} \sum_{k=1}^{M} x_{ik}|k\rangle$ where x_{ik} is the k-th entry of x_i. Suppose $M = 2^m$ is a power of 2. Following the setup in block-encodings and [431, Lemma 47], we must define sets of M orthonormal states $\{|\psi_i\rangle\}$ and $\{|\phi_j\rangle\}$. We choose $|\psi_i\rangle = (\|x_i\||x_i\rangle + \sqrt{1 - \|x_i\|^2}|M+1\rangle)(H^{\otimes m}|i\rangle)|0^m\rangle$, where H denotes the Hadamard transform. We choose $|\phi_j\rangle = (\|x_j\||x_j\rangle + \sqrt{1 - \|x_j\|^2}|M+2\rangle)|0^m\rangle(H^{\otimes m}|j\rangle)$. These states can be prepared in $O(\log(M))$ depth using $O(M)$ total gates and ancilla qubits with methods for controlled state preparation from classical data. It can be verified that these sets are orthonormal, and that $\langle\psi_i|\phi_j\rangle = \langle x_i, x_j\rangle/M$. Hence, the Gram matrix construction yields a block-encoding of K/M with normalization factor 1.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$\widetilde{O}(\alpha\kappa_A/\epsilon')$ (see Chapter 18 on QLSSs). We may also note that as $u = A^{-1}v$ and $\|v\| = 1$, we have $\|u\| \leq \kappa_A/\|A\|$. Thus, the overall circuit depth required to classify x_* is

$$\widetilde{O}\left(\frac{\alpha\kappa_A^2}{\|A\|^2}\right).$$

There is no explicit poly(N, M) dependence. However, for certain datasets and parameter choices, such dependence could be hidden in κ_A or α, making an apples-to-apples comparison with Gaussian elimination less clear.

Furthermore, this task has been dequantized under the assumption of SQ access [350, 271, 924]. In time scaling as poly$(\|A\|_F, \epsilon^{-1}, \log(NM))$, one can classically sample from the solution vector $|u\rangle$ to error ϵ, and furthermore, one can estimate inner products $\langle\tilde{u}|\tilde{v}\rangle$ in time $O(1/\epsilon^2)$ [976].[7] However, the cost can be reduced through a trick that is analogous to how the quantum algorithm can block-encode the $\gamma^{-1}I$ part of A separately to avoid the dependence on a large $\|A\|_F$. In particular, [271, Corollary 6.18] gives a classical complexity that would be polynomially related to the quantum complexity above under appropriate matching of parameters, but the power of this polynomial speedup could still be significant. In any case, such a speedup crucially requires log-depth QRAM access to the training data, which requires total gate complexity $\Omega(NM)$ and $O(NM)$ ancilla qubits.

Example 3: Supervised cluster assignment

Actual end-to-end problem: Suppose we are given access to a vector $x \in \mathbb{C}^N$ and a set of M samples $\{y_j \in \mathbb{C}^N\}_{j=1,\ldots,M}$. We want to estimate the distance between x and the centroid of the set $\{y_j\}$ to judge whether x was drawn from the same set as $\{y_j\}$. If we have multiple sets $\{y_j\}$, we can infer that x belongs to the one for which the distance is shortest; as a result, this is also called the "nearest-centroid problem." Specifically, the computational task is to estimate $\|x - \frac{1}{M}Y\mathbf{1}\|$ to additive constant error ϵ with probability $1 - \delta$, where $Y \in \mathbb{C}^{N\times M}$ is the matrix whose columns are y_j, and $\mathbf{1}$ is the vector of M ones—the vector $Y\mathbf{1}/M$ is the centroid of the set.

Dominant resource cost/complexity: Naively computing the centroid incurs classical cost $O(NM)$. In [707], a quantum solution to this problem was proposed. Let $\bar{x} = x/\|x\|$ and let $\bar{Y}$ be normalized so that all columns have unit

[7] The method of doing so is succinct to describe (see, e.g., [978]). First, one uses sample access to the vector $\tilde{u}$ to generate an index i at random, with probability $|\tilde{u}_i|^2/\|\tilde{u}\|^2$. Then, one uses query access to $\tilde{u}$ and $\tilde{v}$ to compute the quantity $\mathcal{R} = (\tilde{v}_i\|\tilde{u}\|)/(\tilde{u}_i\|\tilde{v}\|)$. The expectation value of $\mathcal{R}$ is precisely $\langle\tilde{u}|\tilde{v}\rangle$, and the variance is upper bounded by 1. Thus, an estimate of $\langle\tilde{u}|\tilde{v}\rangle$ to ϵ precision is obtained by averaging $O(1/\epsilon^2)$ samples of the random variable $\mathcal{R}$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

norm. Define the $N \times (M+1)$ matrix R and length-$(M+1)$ vector w as follows:

$$R = \begin{pmatrix} \bar{x} & \bar{Y}/\sqrt{M} \end{pmatrix}, \qquad w = \begin{pmatrix} \|x\| \\ -1_Y/\sqrt{M} \end{pmatrix},$$

where 1_Y is the length-M vector containing the norms of the columns of Y, defined such that $\bar{Y}1_Y = Y\mathbf{1}$. Then, $Rw = x - \frac{1}{M}Y\mathbf{1}$. Using methods for block-encoding and state preparation from classical data, one constructs $O(\log(NM))$-depth circuits that block-encode R (with normalization factor $\|R\|_F = 2$) and prepare the state $|w\rangle$. If we apply the block-encoding of R to $|w\rangle$ and measure the block-encoding ancillas, the probability that we obtain $|0\rangle$ is precisely $(\|Rw\|/2\|w\|)^2$. Thus, using amplitude estimation, one learns $\|Rw\|$ to precision ϵ with probability at least $1-\delta$ at cost $O(\|w\| \log(1/\delta)/\epsilon)$ calls to the log-depth block-encoding and state preparation routines.

The advantage over naive classical methods essentially boils down to the assumption of efficient classical data loading for a specific dataset. Subsequently, this quantum algorithm was dequantized, and it was understood that a similar feat is possible classically in the SQ access model [977]. Specifically, the classical algorithm runs in time $\widetilde{O}(\|w\|^2 \log(1/\delta)/\epsilon^2)$, reducing the exponential speedup to merely quadratic.

Caveats

The overwhelming caveat in these and other proposals is access to the classical data in quantum superposition. These quantum machine learning algorithms assume that we can load a vector of N entries or a matrix of N^2 entries in polylog(N) time. While efficient quantum data structures, that is, QRAM, have been proposed for this task, they introduce a number of caveats. In order to coherently load N pieces of data in $\log(N)$ time, QRAM uses a number of ancilla qubits, arranged in a tree structure. To load data of size N, the QRAM data structure requires $O(N)$ qubits, which is exponentially larger than the $O(\log(N))$ data qubits used in the algorithms above. This spatial complexity does not yet include the overheads of quantum error correction and fault-tolerant computation, in particular the large spatial resources required to distill magic states in parallel. As such, we do not yet know if it is possible to build a QRAM that can load the data sufficiently quickly, while maintaining moderate spatial resources.

In addition, achieving speedups by efficiently representing the data as a quantum state may suggest that classical methods based on tensor networks could achieve similar performance, in some settings. Taking this line of reasoning to the extreme, a number of efficient classical algorithms have been developed by "dequantizing" the quantum algorithms. That is, by assuming an

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

analogous access model (the SQ access model) to the training data, as well as some assumptions on sparsity and/or rank of the inputs, there exist approximate classical sampling algorithms with polynomial overhead as compared to the quantum algorithms [977, 976]. This means that any apparent exponential speedup must be an artifact of the data loading/data access assumptions.

A further caveat is inherited from the QLSS subroutine, which is that the complexity is large when the matrices involved are ill conditioned. This caveat is somewhat mitigated in the Gaussian process regression and support vector machine examples above, where the matrix to be inverted is regularized by adding the identity matrix.

End-to-end resource analysis

To the best of our knowledge, full end-to-end resource estimation has not been performed for any specific quantum machine learning tasks.

Outlook

Much of the promise of quantum speedup for classical machine learning based on linear algebra hinges on the extent to which quantum algorithms can be dequantized. At present, the results of [977] seem to prohibit an exponential speedup for many of the problems proposed, but there is still the possibility of a large polynomial speedup. The most recent asymptotic scaling analysis [271] for dequantization methods still allows for a power 4 speedup in the Frobenius norm of the "data matrix" and a power 11 speedup in the polynomial approximation degree (see [86] for more details). However, the classical algorithms are steadily improving, and their scaling might be further reduced.

It is also worth noting that the classical probabilistic algorithms based on the SQ access model are not currently used in practice. This could be due to a number of reasons, including the poor polynomial scaling, the fact that the access model might not be well suited to many practical scenarios, or simply because the method is new and has not been tested in practice (see [61, 270] for some work in this direction).

On the other hand, some machine learning tasks based on quantum linear algebra are not known to be dequantized, such as Gaussian process regression under the assumption that the kernel matrix is sparse. In particular, avoiding dequantization and achieving an exponential quantum speedup appears to require that the matrices involved are simultaneously sparse, high rank, and well conditioned.[8] In this situation, quantum algorithm can leverage block-encodings

[8] Dequantization can also be avoided even when the matrices involved are dense, provided that they are given by a product of a small number of sparse matrices. For example, it was described in [1064] how an exponential speedup may be possible for a certain ML task, where

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

for which the normalization factor is equal to the sparsity, rather than general block-encodings of classical data for which the normalization factor is the Frobenius norm. The complexity of quantum-inspired classical algorithms based on SQ access will still grow polynomially with the Frobenius norm even when the matrices are sparse,[9] although other classical algorithms may be able to exploit the sparsity more directly. Perhaps unsurprisingly, the prototypical matrices that satisfy these criteria are sparse unitary matrices, such as those naturally implemented by a local quantum gate. For unitary matrices, the condition number is 1, and the Frobenius norm is equal to the square root of the Hilbert space dimension—exponentially large in the system size. (As a simple example, consider the identity matrix on n qubits.) A central question is whether situations like this occur in interesting end-to-end machine learning problems. Even if they do, an exponential speedup is not guaranteed. An additional hurdle arises in the quantum readout step, which incurs a cost scaling as the inverse in the precision target. To avoid exponential overhead, the end-to-end problem must not require exponentially small precision.

Further reading

For further reading on specific machine learning tasks where quantum algorithms have been proposed, we refer the reader to [151, 293, 916]. We have not covered dequantization techniques in great detail; for an accessible summary and perspective, see [978], and for a more detailed overview, see [979].

9.2 Quantum machine learning via energy-based models

Overview

An important class of models in machine learning is *energy-based models*, which are heavily inspired by statistical mechanics. The goal of energy-based

the matrix to be inverted is neither sparse nor low rank; rather, it is related to a sparse matrix via the discrete Fourier transform (a dense unitary matrix). A block-encoding for the relevant matrix is constructed by leveraging the quantum Fourier transform (QFT). Note that the QFT can be decomposed into a product of sparse matrices corresponding to the local unitary gates in the quantum circuit for QFT.

[9] For example, consider the complexity of algorithms for pseudoinversion of an s-sparse matrix A. Let the rank of A be r, which may be as large as the matrix dimension, and suppose all nonzero singular values of the matrix lie in the interval $[1/\kappa, 1]$. This implies that $\sqrt{r}/\kappa \leq \|A\|_F \leq \sqrt{r}$, and thus $\kappa\|A\|_F \geq \sqrt{r}$. As a consequence, the complexity of quantum-inspired algorithms with SQ access in [271, 433, 924]—scaling as $\mathrm{poly}(\|A\|_F, \kappa, 1/\epsilon)$—will necessarily grow as a polynomial of the matrix rank, even for fixed sparsity. On the other hand, the complexity of the quantum algorithm with QRAM that applies the QLSS and amplitude estimation—scaling as $\mathrm{poly}(s\|A\|_{\max}, \kappa, 1/\epsilon)$—is independent of the matrix rank for well-conditioned A (i.e., $\kappa = O(1)$).

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

models is to train a physical model (i.e., tune the interaction strengths between a set of particles) such that the model closely matches the training set when the model is in thermal equilibrium (made more precise below). Energy-based models are an example of *generative* models since, once they are trained, they can then be used to form new examples that are similar to the training set by sampling from the model's thermal distribution.

Due to their deep connection to physics, energy-based models are prime candidates for various forms of quantization. However, one challenge faced by quantum approaches is that the statistical mechanical nature of the learning problem also often lends itself to efficient, approximate classical methods. As a result, the best quantum algorithms may also be heuristic in nature, which prevents an end-to-end complexity analysis. While energy-based models are less widely used than deep neural networks today, they were an important conceptual development in machine learning [893] and continue to foster interest due to their sound theoretical basis, and their connection to statistical mechanics.

There are a number of proposals for generalizing energy-based models to quantum machine learning. The starting point is a graph where the vertices are divided into *visible* $\{v\}$ and *hidden* $\{h\}$ nodes. When each node is assigned a value in some discrete or continuous set, this constitutes a "configuration" (h, v) of the model. A training set $\mathcal{D}$ is provided as input, containing a list of configurations of the visible vertices. The hidden nodes are not part of the training set, but including them is essential for the model to be able to capture latent variables in the data.

A graphical model is then built on the vertices—each vertex is a physical system (such as a spin-1/2 particle) and edges between vertices represent physical interactions. The model is described by an energy functional $H(h, v)$, which assigns an energy value to each possible configuration (h, v) of the vertices. For example, in Boltzmann machines (BMs), the vertices are assigned binary variables, and the interactions are Ising interactions. The model can be used to generate samples (e.g., via Markov chain Monte Carlo methods) from the thermal distribution (also known as the Boltzmann distribution or the Gibbs distribution) at unit temperature, that is, the distribution where each configuration (h, v) is sampled with probability proportional to $\mathrm{e}^{-H(h,v)}$. In unsupervised learning tasks, provided a set of training samples of configurations of the visible units v, the goal is to tune the interaction weights of the model such that the model's thermal distribution best matches the distribution that generated the training set.

Quantum algorithms can potentially be helpful for training classical graphical models. One can also generalize the model itself by allowing the physical

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

systems on each vertex to be quantum, and interactions between systems to be noncommuting.

Actual end-to-end problem(s) solved

Classical graphical models: Let $G = (V, E)$ denote a graph with vertices V and edges E. For classical models, each vertex j is assigned a binary variable $z_j = \pm 1$. The variables are split into visible and hidden nodes, $z \in \{v\} \cup \{h\}$. For classical BMs, the energy functional is often taken to be quadratic[10] with weights $\{b_i, w_{ij}\}$:

$$H(z) = \sum_{i \in V} b_i z_i + \sum_{(i,j) \in E} w_{ij} z_i z_j. \tag{9.6}$$

Note that interactions can occur between any pair of nodes (hidden or visible). In the special case of a restricted Boltzmann machine (RBM), each edge must pair up a hidden node with a visible node (i.e., the graph is bipartite). This restriction makes the model less expressive than graphs with edges between hidden nodes, but it leads to simplifications for certain training approaches.

The thermal distribution corresponding to the energy functional (at unit temperature) associates each configuration v of visible nodes with a probability $p(v)$ such that

$$p(v) = \sum_h p(h, v), \qquad p(h, v) = \frac{\mathrm{e}^{-H(h,v)}}{\mathcal{Z}}, \qquad \mathcal{Z} = \sum_{h,v} \mathrm{e}^{-H(h,v)},$$

where $\mathcal{Z}$, the partition function, is the normalization to ensure probabilities sum to 1. Even though hidden nodes are integrated out in the calculation of $p(v)$, they impact the distribution of $p(v)$ through their interactions with the visible nodes.

Given a training set $\mathcal{D} = \{v_1, v_2, \ldots, v_{|\mathcal{D}|}\}$ of sample configurations of the visible nodes, the goal of the training phase is to modify the weights $\theta \in \{b_i\} \cup \{w_{ij}\}$ such that samples from the thermal distribution of the model most closely match the training samples. Ideally, this is done by finding the set of weights that maximizes the likelihood of observing the samples, that is, $\prod_{v \in \mathcal{D}} p(v)$, or, equivalently, minimizing the (normalized) log-likelihood loss function, defined as

$$L(b, w) = -\frac{1}{|\mathcal{D}|} \sum_{v \in \mathcal{D}} \log(p(v)). \tag{9.7}$$

[10] This quadratic energy functional is related to the Sherrington–Kirkpatrick (SK) model [933] with an external field, which is a model for spin glasses in the statistical mechanics literature. For the SK model, the couplings w_{ij} are chosen randomly for each pair of nodes, and it is typically computationally hard to find configurations with optimal energy (see Section 4.2 on beyond quadratic speedups in combinatorial optimization for additional information).

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

The loss function can be minimized using some variant of gradient descent, which requires the evaluation of the derivatives $\partial_\theta L$ for $\theta \in \{b_i\} \cup \{w_{ij}\}$. For the energy functional above, these derivatives can be readily calculated from ensemble averages (see, e.g., [1041]). For example,

$$\frac{\partial L}{\partial w_{ij}} = \langle z_i z_j \rangle_{v \in \mathcal{D}} - \langle z_i z_j \rangle \,, \tag{9.8}$$

where $\langle \cdot \rangle$ denotes an average over samples from the thermal distribution $p(h, v)$, while $\langle \cdot \rangle_{v \in \mathcal{D}}$ denotes an average where v is drawn at random from the training set $\mathcal{D}$, and h is sampled from the thermal distribution conditioned on that choice of v. Without any further restrictions, the gradients will typically be difficult to evaluate (or even to estimate accurately). An exact computation requires computing a sum over the exponential number of configurations of the vertices.

In some cases, good estimates of the gradients can be obtained by repeatedly drawing samples from the thermal distribution and computing averages. Samples can be generated with Markov chain Monte Carlo (MCMC) methods such as the Metropolis–Hastings algorithm or simulated annealing; however, the time required to sample from a distribution close to the thermal distribution depends on the mixing time of the Markov chain, which is generally unknown and can also be exponential in the graph size. Additionally, many samples need to be generated to produce a robust average, with precision ϵ requiring $O(1/\epsilon^2)$ samples. Approximate classical methods, such as contrastive divergence [531], avoid this issue by initializing the Markov chain at one of the training samples and deliberately taking a small number of steps—this does not exactly correspond to optimizing the log-likelihood but in some cases has empirical success [916]. Indeed, here we see the benefit of restricting to bipartite graphs in RBMs: since there are no edges between hidden nodes, the Gibbs distribution over the hidden nodes is independent from node to node, conditioned on a fixed setting of the visible nodes. This enables a simple exact calculation for the first term of Eq. (9.8), and it is also key to the success of estimating the second term with contrastive divergence, where the hidden layer and the visible layer are conditionally sampled in alternating fashion.

Once the model has been trained, new samples can also be generated via the same MCMC methods. The end-to-end tasks are (i) training the model, and then, (ii) generating samples from the trained model to accomplish some larger machine learning goal.

Quantum graphical models: A separate end-to-end problem is found by generalizing the model itself to be quantum. For example, one can start with a clas-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

sical BM and promote the binary variables to qubits. The energy functional is promoted to a quantum Hamiltonian and augmented with a transverse field, which does not commute with the Ising interactions. The result is a quantum Boltzmann machine (QBM), described by a transverse-field Ising (TFI) Hamiltonian [30] (cf. Eq. (1.4)):

$$H_{\text{QBM}} = -\sum_{i\in V}(\kappa_i X_i + b_i Z_i) - \sum_{(i,j)\in E} w_{ij} Z_i Z_j \,, \tag{9.9}$$

where X_i and Z_i are the Pauli-X and Pauli-Z operators on qubit i, and b_i, κ_i, w_{ij} are real variational parameters of the model. The ground or Gibbs state of H_{QBM} can be prepared in a variety of ways, including the adiabatic algorithm, Gibbs sampling, or as a variational quantum algorithm. These states can be measured (in the Z basis or in the X basis), yielding samples of the variables v, h drawn from different distributions than the thermal distribution for the classical BM. Alternatively, one can trace out the hidden nodes, viewing the leftover quantum state on the visible nodes as the output of the QBM; this may be suitable if the input data is also quantum. As in the classical case, the training phase for a QBM consists of varying the weights via gradient descent to maximize a likelihood function. However, the noncommutativity of the Hamiltonian leads to complications: the gradients of the loss function are no longer directly given by sample expectation values as was the case in Eq. (9.8), although workarounds have been proposed [30, 614, 1037, 40, 1093]. For example, in the case of classical input data, sample expectation values can be used to optimize function that is not equal to the loss function, but can be shown to be an upper bound on it using the Golden–Thompson inequality [30]. Alternatively, in the case of quantum input data, and assuming the QBM has no hidden nodes, the relevant loss metric is the relative entropy and its gradients can again be related to sample averages [614]—this scenario is closely related to the Hamiltonian learning problem. In any case, the end-to-end problem is to train these models and generate samples.

Dominant resource cost/complexity

Complexity of classical graphical models: Recall that for classical BMs, one wishes to produce samples from the thermal distribution corresponding to the energy functional in Eq. (9.6), that is, Gibbs sampling (of diagonal Hamiltonians), either to assist in training the model or, if it has already been trained, to make inferences or generate new data. Specifically, given $H(h, v)$, one wishes to draw samples of (h, v) with probability proportional to $\mathrm{e}^{-H(h,v)}$, either with v free or with v fixed (sometimes referred to as "clamped") to a particular value from the training set $\mathcal{D}$. Classically, one approach is simulated annealing or

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

other MCMC algorithms. Quantumly, one can take one of several analogous approaches, including "quantum simulated annealing" [944] and quantum annealing, discussed as follows.

For quantum simulated annealing, one prepares the coherent Gibbs state $\sum_{v,h} \sqrt{p(h,v)}|v,h\rangle$, and a quadratic speedup is obtained over classical simulated annealing. The method is to construct a Hamiltonian whose ground state is the coherent Gibbs state at temperature T (i.e., for which probabilities $p(h,v)$ are proportional to $e^{-H(h,v)/T}$), and follow an adiabatic path from $T = \infty$ to $T = 1$. Following the path is accomplished by repeatedly performing quantum phase estimation (QPE) to project onto the ground state of the Hamiltonian at a given temperature. As is typical for the adiabatic algorithm, the cost of this procedure is dominated by the inverse of the spectral gap—this is the precision required for QPE to succeed. Specifically, for a graphical model with $|V|$ vertices, the runtime will be $\text{poly}(|V|)/\Delta$, where Δ is the minimum spectral gap. Importantly, Δ can be related to the maximum mixing time t_{mix} of the simulated annealing Markov chain, as $1/\Delta = O(\sqrt{t_{\text{mix}}})$, which leads to the quadratic speedup, although it is possible that Δ is exponentially small in $|V|$.

An alternative method for preparing (and sampling from) the coherent Gibbs state was proposed in [1041]. There, one begins in an easy-to-prepare coherent mean-field state approximating the coherent Gibbs state. Then, one performs rejection sampling with amplitude amplification to gain a quadratic speedup over the analogous classical method. Additionally, it was proposed to use amplitude estimation to gain a quadratic improvement in the number of samples needed to achieve precision ϵ, from $O(1/\epsilon^2)$ to $O(1/\epsilon)$, mirroring later analyses that work for general quantum-accelerated Monte Carlo methods [773]. If these $O(1/\epsilon)$ quantum samples are each for the same training sample $v \in \mathcal{D}$, this is straightforward; however, if the samples are drawn randomly from $v \in \mathcal{D}$, achieving the quadratic speedup from amplitude estimation requires accessing the data in $\mathcal{D}$ coherently and quickly. Such data access is provided by the quantum random access memory (QRAM) primitive, for which the circuit *depth* can be logarithmic in the size of the training data, at the expense of a number of ancilla qubits (and total gates) that is linear in the size of the training data.

For quantum annealing, the idea is to add a uniform transverse field, as in the QBM of Eq. (9.9) with $\kappa_i = \kappa_j$ for all i, j. The transverse field is initially strong, and slowly turned off. This is similar to the adiabatic algorithm, but differs in that it is specifically carried out at finite ambient temperature. Thus, the system-bath interaction of the device naturally drives the state to the Gibbs state, which coincides with the classical thermal distribution once the transverse field is turned off. This is a heuristic method; it is efficient but there are

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

few success guarantees. The hope is that the inclusion of an initial transverse field induces nonclassical fluctuations that help the system avoid becoming trapped in local minima as the transverse field is turned off.

Overall, computing the gradient of the loss function with respect to one parameter, up to precision ϵ, will require complexity $O(S/\epsilon)$, where S is the complexity of sampling from the Gibbs state. The above assumes log-depth QRAM to be able to estimate the $\langle z_i z_j \rangle_{v\in\mathcal{D}}$ term of Eq. (9.8). The complexity of S will be poly($|V|$) $\sqrt{t_{\text{mix}}}$ if a quantum simulated annealing approach is used, or some hard-to-analyze quantity if the quantum annealing approach is used. If the number of training samples is small, one can also sequentially compute the sum over $v \in \mathcal{D}$ and avoid the assumption of log-depth QRAM, leading to complexity $O(S|\mathcal{D}|/\epsilon')$ (where $\epsilon' \geq \epsilon$ may be order-1). This must be carried out for all $|E| + |V|$ weights in the model, although these could be simultaneously estimated to precision ϵ at cost $\widetilde{O}(\sqrt{|E| + |V|}/\epsilon)$ samples, using methods from [549], which leverage the quantum gradient estimation primitive. It is not clear what value of ϵ is required in practice. Reference [1041] takes $\epsilon \sim 1/\sqrt{|\mathcal{D}|}$, to match the natural uncertainty coming from a finite number of training samples. In this case, the overall complexity is dominated by

$$\widetilde{O}\left(S \cdot \sqrt{|V| + |E|} \cdot \sqrt{|\mathcal{D}|}\right) \tag{9.10}$$

assuming log-depth QRAM, and

$$\widetilde{O}\left(S \cdot \sqrt{|V| + |E|} \cdot |\mathcal{D}|\right) \tag{9.11}$$

without log-depth QRAM (the precision for each training sample can be taken as $\epsilon' = O(1)$). The linear dependence on $|\mathcal{D}|$ could potentially be mitigated by first classically computing a "core set" $\mathcal{D}'$ satisfying the requirements that $|\mathcal{D}'| \ll |\mathcal{D}|$ and that replacing $\mathcal{D}$ with $\mathcal{D}'$ causes minimal change to the loss function in Eq. (9.7) [498].

Complexity of quantum graphical models: For QBMs, the dominant cost comes from producing samples from the quantum Gibbs state for the system in Eq. (9.9), that is, the state $\rho \propto e^{-H_{\text{QBM}}}$, which can be accomplished through methods for Gibbs sampling. Rigorous methods for Gibbs sampling may scale exponentially in the size of the graph, without further assumptions. Such scaling would likely not be tolerable in practice. However, Monte Carlo–style methods for Gibbs sampling, which follow a similar approach as MCMC, but in an inherently quantum way, may be more effective in this case. These could have poly($|V|$) scaling for some parameter settings, but must also have exponential scaling in the worst case, as sampling low-energy Ising-model configurations is known to be NP-hard.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

One can also heuristically apply quantum annealing, beginning from a large transverse field and reducing its strength slowly to some final nonzero value. However, some hardware platforms may only admit global control over the transverse field, preventing one from tuning the transverse-field strengths κ_i individually. In any of these approaches, it is difficult to make any rigorous statements about the Gibbs sampling complexity.

Existing resource estimates

There are no logical resource estimates for quantum annealing. However, [7, 116] discuss in detail how to embed the fully connected architecture of a RBM into the 2D lattice architecture available on planar quantum annealers. Reference [116] reports an embedding ratio scaling which is roughly quadratic—that is, a graphical model with $|V|$ vertices requires $O(|V|^2)$ qubits to accommodate the architectural limitations of the device. A proper resource estimation has not been performed for the fault-tolerant algorithm of [1041].

Caveats

There are two main caveats to quantum approaches to training classical models, which apply to both the annealing and to the fault-tolerant setting. First, classical heuristic algorithms, such as greedy methods or contrastive divergence, often perform well in practice and are the method of choice for existing classical analyses. These methods are also often highly parallelizable. If the quantum algorithm offers a speedup over a slower, exact classical method, this may not be relevant if the faster approximate classical methods are already sufficient. Second, the situations where one might hope for the heuristic quantum annealing approach to perform better might not be relevant problems, for instance, in highly regular lattice-based problems.

A caveat of the QBM is that the gradients of the loss function are not exactly related to sample averages, and imperfect workarounds, such as those proposed in [30], must be pursued. Like many other situations in machine learning, the resulting end-to-end solution is heuristic and evidence of its efficacy requires empirical demonstration.

Comparable classical complexity and challenging instance sizes

For classical models, an exact computation of the gradients would scale exponentially in the size of the graph, as $O(|\mathcal{D}|2^{|V|})$ for the gradient of a single parameter. Approximate methods based on simulated annealing or other MCMC methods would scale as $O(S_c/\epsilon^2)$, where S_c is the classical sample time, scaling as $S_c = \text{poly}(|V|)t_{\text{mix}}$. On the other hand, these methods can also be implemented heuristically at reduced cost (e.g., contrastive divergence, where

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

one does not wait for the chain to mix), and they can also be implemented on parallel architectures. For instance, in [618], an architecture was proposed to train deep BMs efficiently. Experiments demonstrated that heuristic training methods could be carried out for graphs of size 1 million in 100 seconds on field-programmable gate arrays available in 2010. Much larger sizes would be accessible to a scaled-up version of the same architecture on modern hardware. It is unlikely that any exact method, quantum or classical, could match this efficiency.

For the quantum models, the classical complexity of sampling from the Gibbs state of the model would be exponential in the graph size $|V|$. Thus, training these models would likely not be pursued classically.

Speedup

For the classical models, the speedup can be quadratic in most of the parameters: producing a sample can in some cases be sped up quadratically, and the number of samples required to achieve a certain precision also enjoys a quadratic speedup (e.g., t_{mix} to $\sqrt{t_{\text{mix}}}$ and $O(1/\epsilon^2)$ to $O(1/\epsilon)$). The methods that give these provable quadratic speedups are based on primitives such as amplitude amplification, where superquadratic speedups are not possible without exploiting additional structure. Larger superpolynomial speedups are only possible under optimistic assumptions about the success of heuristic quantum annealing approaches at producing samples faster than classical approaches.

For the quantum models, the speedup is technically exponential, assuming that for the models considered, quantum algorithms for Gibbs sampling scale efficiently while approximate classical methods (e.g., tensor networks) scale exponentially. Indeed, it was shown in [1042] that certain QBMs are BQP-complete, in the sense that its ground state can be efficiently prepared on a quantum computer, and any quantum computation (including those with exponential speedup) can be encoded into its ground state. However, this construction is artificial, and it has yet to be demonstrated that there are specific real-world machine learning tasks where these models offer a speedup over the best available classical machine learning model for the same task.

Outlook

While energy-based models are naturally in a form that can readily be extended to the quantum domain, there still lacks decisive evidence of quantum advantage for a specific end-to-end classical machine learning problem. There remains some uncertainty on the outlook of these approaches due to the centrality of heuristic quantum approaches. One may hold out hope that these heuristics could outperform classical heuristics in some specific settings, but

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the success of classical heuristics and effectiveness of approximate classical approaches present a formidable barrier to achieving any quantum advantage in this area.

Further reading

We refer the reader to [916] for more information on quantum approaches to energy-based models.

9.3 Tensor PCA

Overview

Inference problems play an important role in machine learning. One of the most widespread methods is principal component analysis (PCA), a technique that extracts the most significant information from a stream of potentially noisy data. In the special case where the data is generated from a rank-1 vector plus Gaussian noise—the spiked matrix model—it is known that there is a phase transition in the signal-to-noise ratio [536]: above the transition point, the principal component can be recovered efficiently, while below the transition point, the principal component cannot be recovered at all. In the tensor extension of the problem, there are two transitions. One information theoretical, below which the principal component cannot be recovered, and another computational, below which the principal component can be recovered, but only inefficiently, and above which it can be recovered efficiently. Thus, the tensor PCA problem offers a much richer mathematical setting, which has connections to optimization and spin glass theory; however, it is yet unclear if the tensor PCA framework has natural practical applications. A quantum algorithm [509] for tensor PCA was proposed which has provable runtime guarantees for the spiked tensor model; it offers a potentially *quartic* speedup over its classical counterpart and also efficiently recovers the signal from the noise at a smaller signal-to-noise ratio than other classical methods. This algorithm was further developed in [904] and extended to also give a quartic speedup for a related discrete optimization problem called "planted noisy kXOR," which is argued to have possible relevance in cryptography.

Actual end-to-end problem(s) solved

Consider the spiked tensor problem. Let $v \in \mathbb{R}^N$ (or $\in \mathbb{C}^N$)[11] be an unknown signal vector, and let $p \in \mathbb{N}$ be a positive integer. Construct the tensor

$$T = \lambda v^{\otimes p} + V,$$

[11] Reference [509] provides reductions between real and complex cases.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

where V is a random tensor in $\mathbb{R}^{N^p}$ (or $\mathbb{C}^{N^p}$), with each entry drawn from a normal distribution with mean 0 and variance 1. The vector v is assumed to have norm $\sum_j v_j^* v_j = \sqrt{N}$ and can be identified with a quantum state. The quantity λ is the signal-to-noise ratio.

The main question we are interested in is for what values of λ can we detect or reconstruct v from (full) access to T, and how efficiently can this be done? In [874], it was shown that the maximum likelihood solution w^{ML} to the objective function

$$w^{\mathrm{ML}} = \operatorname*{argmax}_{w \in \mathbb{C}^n} \langle T, w^{\otimes p} \rangle$$

will have high correlation with v as long as $\lambda \gg N^{(1-p)/2}$, where $\langle \cdot, \cdot \rangle$ denotes the standard dot product after writing the N^p entries of the tensor as a vector. However, the best known *efficient* classical algorithm [1033] requires $\lambda \gg N^{-p/4}$ to recover an approximation of v. Using the spectral method, that is, mapping the tensor T to a $N^{p/2} \times N^{p/2}$ matrix and extracting the maximal eigenvalue, recovery can be done in time complexity $O(N^p)$, ignoring logarithmic prefactors.

Hastings [509] proposes classical and quantum algorithms to solve the spiked tensor model by first mapping T to a bosonic quantum Hamiltonian with N modes, n_{bos} bosons, and p-body interactions, where n_{bos} is a tunable integer parameter satisfying $n_{\mathrm{bos}} > p/2$

$$H_{\mathrm{PCA}}(T) = \frac{1}{2}\left(\sum_{\mu_1,\ldots,\mu_p=1}^{N} T_{\mu_1,\ldots,\mu_p} \left(\prod_{i=1}^{p/2} a_{\mu_i}^\dagger \right) \left(\prod_{j=1+p/2}^{p} a_{\mu_j} \right) + \mathrm{h.c.} \right), \tag{9.12}$$

where h.c. stands for Hermitian conjugate. Here, the operators a_μ and $a_\mu^\dagger$ are annihilation and creation operators of a boson in mode μ, and we restrict to the sector for which $\sum_\mu a_\mu^\dagger a_\mu = n_{\mathrm{bos}}$. As n_{bos} increases, the number of particles increases and the complexity of the algorithm grows, but the threshold for λ above which recovery is possible also decreases.

Hastings shows that the vector v can be efficiently recovered from a vector in the large energy subspace of $H_{\mathrm{PCA}}(T)$ when the largest eigenvalue of $H_{\mathrm{PCA}}(T)$ is at least a constant factor larger than $E_{\max}$, where $E_{\max}$ corresponds to the case where there is no signal. It is shown that, roughly,

$$E_{\max} \sim n_{\mathrm{bos}}^{p/4+1/2} N^{p/4},$$
$$E_0 \approx \lambda (p/2)! \binom{n_{\mathrm{bos}}}{p/2} N^{p/2} \approx \lambda n_{\mathrm{bos}}^{p/2} N^{p/2},$$

where E_0 is the maximum eigenvalue of $H_{\mathrm{PCA}}(T)$. Thus, if $\lambda \gg N^{-p/4}$, there will be a gap between E_0 and $E_{\max}$, and this gap grows as n_{bos} increases. This

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

enables the quantum algorithm to recover the signal even for signal strength as weak as $\lambda \gg N^{-p/4}$.

Hastings considers the case where p is constant and N grows, and assumes that $n_{\text{bos}} = O(N^\theta)$ for some p-dependent constant $\theta > 0$ chosen sufficiently small. In fact, ultimately, it is determined that in the recovery regime $\lambda \gg N^{-p/4}$, the parameter n_{bos} need only scale as $\text{polylog}(N)(N^{-p/4}/\lambda)^{4/(p-2)}$. In any case, terms in the complexity $O(N^p)$ are dominated by terms $O(N^{n_{\text{bos}}})$.

The idea of mapping the order-p tensor T to an order-2 tensor (i.e., a matrix) H_{PCA}, and then solving the planted inference problem by (classically) extracting spectral information from H_{PCA}, was also independently discovered in [1033], where it is called the "Kikuchi method." There, a tunable parameter ℓ plays the role of n_{bos}, although the two formulations offer different intuitions to motivate the method; their relationship is discussed in detail in [904].

Dominant resource cost/complexity

Hastings shows that the dominant eigenvector of H_{PCA} can be classically extracted in $\widetilde{O}(N^{n_{\text{bos}}})$ time via the power method, where the tilde indicates that we ignore polylogarithmic factors.

He proposes three quantum algorithms for the same problem. The first runs quantum phase estimation on a random state. Since the random state will have squared overlap $\Omega(N^{-n_{\text{bos}}})$ with the high-energy subspace, the expected number of repetitions of phase estimation is $O(N^{n_{\text{bos}}})$. The second algorithm proposes to further use amplitude amplification, reducing the complexity to $O(N^{n_{\text{bos}}/2})$. The third algorithm further improves the complexity by choosing a specific initial high-energy state, and showing that the overlap with the state scales as $\Omega(N^{-n_{\text{bos}}/2})$, which combined with amplitude amplification, leads to a $O(N^{n_{\text{bos}}/4})$ complexity. As discussed above, the estimates assume that factors of $O(N^p)$ can be ignored, since they are negligible with respect to the query complexity of $N^{O(n_{\text{bos}})}$.

This constitutes a quartic speedup over the classical spectral algorithm acting on H_{PCA} for the same choice of n_{bos} that is also presented in [509]. Since the ansatz state is a product state, it can be prepared efficiently.

Hastings further argues that the Hamiltonian simulation of H_{PCA} within the phase estimation subroutine can be accomplished by viewing H_{PCA} as a sparse or local matrix. Specifically, we can view H_{PCA} as a second-quantized Hamiltonian with N registers storing a $O(\log(n_{\text{bos}}))$-bit number representing the occupancy of each mode. The total number of qubits needed is $O(N \log(n_{\text{bos}}))$. Each of the $O(N^p)$ terms in Eq. (9.12) corresponds to a single nonzero entry of one row of H_{PCA} in this basis, so the $O(n_{\text{bos}}^N) \times O(n_{\text{bos}}^N)$ matrix H_{PCA} is $O(N^p)$-sparse. Alternatively, a more compact representation would be to view H_{PCA} in

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

a first-quantized picture, allocating n_{bos} registers each storing a $O(\log(N))$ bit number corresponding to which mode each of the n_{bos} bosons are in—the total number of qubits needed is $O(n_{\text{bos}} \log(N))$. In order to enforce permutational symmetry, each term in Eq. (9.12) would decompose into $(p/2)!\binom{n_{\text{bos}}}{p/2}$ separate terms that act locally on $p/2$ of the registers. This is closer to the approach taken in the quantum algorithm of [904].

Either way, we can efficiently perform Hamiltonian simulation (e.g., by first constructing a block-encoding of the sparse or local Hamiltonian) to perform quantum phase estimation at total gate complexity $\text{poly}(N, n_{\text{bos}})$ (here interpreting $p = O(1)$), which is negligible compared to $N^{n_{\text{bos}}}$.

Caveats

The spiked tensor model does not immediately appear to be related to any practical problems. Additionally, efficient recovery requires that the signal-to-noise ratio be rather high, which may not occur in real-world settings, and when it does, it is not clear that formulating the problem as a tensor PCA problem will be the most efficient path forward. Relatedly, while the runtime of the algorithm scales subexponentially in N, for large values of n_{bos}, its N dependence of $O(N^{n_{\text{bos}}})$ may still lead to a practically intractable algorithm.

Comparable classical complexity and challenging instance sizes

The algorithms proposed in [509] (see also [1033, 904]) improve on other spectral methods for the spiked tensor model, whenever $n_{\text{bos}} > p/2$ for sufficiently large p. The threshold for which the new algorithms beat the older ones decreases as n_{bos} increases, although the complexity of the algorithm increases with n_{bos}.

Speedup

The quartic speedup over the classical power method is achieved by combining a quadratic speedup from amplitude amplification with a quadratic speedup related to choosing a clever initial state for phase estimation. The existence of the clever initial state is essential for the beyond-quadratic speedup, and it has been related [904] to the BQP-hardness of the guided local/sparse Hamiltonian problem [419].

As discussed above, there is no readout issue, as the vector v can be efficiently recovered from the single particle density matrix obtained from the eigenvector of $H_{\text{PCA}}(T)$. The quantum algorithm has $O(N \log(n_{\text{bos}}))$ space complexity, which is an exponential improvement over the classical spectral algorithm presented in [509] for the same problem. Furthermore, the quartic speedup in time and exponential speedup in space is possible even in the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

absence of a large-scale quantum random access memory (QRAM), since the overall runtime of the algorithm is much larger than the size of the input dataset ($O(N^p)$), and linear (rather than logarithmic) data access cost is tolerable.

In [904], the quartic speedup was extended to apply generally to instances where the "Kikuchi method" is used; improvements to the classical algorithm within this framework would imply commensurate improvements to the quantum algorithm to maintain the quartic relationship.

Outlook

The quartic speedup is very compelling, as beyond-quadratic speedups are rare in quantum algorithms. It is also appealing that the speedup also does not necessarily rely upon an assumption of large-scale QRAM. However, it is currently unclear how applicable this quartic speedup can be in real-world situations.

Further reading

We refer the reader to [904] for a discussion on the intuition and scope of the quartic speedup and additional technical details. We also note that [1091] has studied the quantum approximate optimization algorithm (QAOA) applied to the spiked tensor model, although the result is not directly comparable as it is only shown to succeed for larger values of λ, where additional classical algorithms are also successful.

9.4 Topological data analysis

Overview

In topological data analysis (TDA), we aim to compute the dominant topological features (connected components and k-dimensional holes), known as Betti numbers, of N data points sampled from an underlying topological manifold or of a graph with N vertices. These features may be of independent interest (e.g., the number of connected components in the matter distribution in the universe) or can be used as generic features to compare datasets. We refer to [519] for a recent survey of applications of TDA.

Quantum algorithms for TDA leverage the ability of a register of qubits to efficiently represent a quantum state that stores all cliques in the clique complex built on the topological manifold. The textbook classical algorithm exactly computes the Betti numbers, but its complexity scales polynomially with the number of cliques in the complex, which may grow combinatorially with

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

(N, k). In contrast, quantum algorithms naturally estimate Betti numbers normalized by the number of cliques in the complex, and their complexity scales polynomially in (N, k) for clique-dense complexes. If the error is rescaled so as to be a constant additive error estimate for the Betti number, currently known quantum algorithms provide a quadratic speedup over the best classical algorithms for the equivalent problem. For relative-error estimates of the Betti number, quartic and superpolynomial speedups have been shown for certain families of graphs. In addition, a number of complexity-theoretic results have been shown that provide evidence that estimating normalized Betti numbers is efficient for quantum computers, but classically hard. Nevertheless, finding practical applications for relative error estimates of high-dimensional Betti numbers is an open research question.

Actual end-to-end problem(s) solved

We construct a simplicial complex built from N data points sampled from an underlying manifold. The simplicial complex is a higher-dimensional generalization of a graph, constructed by connecting data points within a given distance of each other. A simplicial complex constructed in this way is known as a clique complex or a Vietoris–Rips complex. We can consider a sequence, known as a filtration, of complexes constructed by connecting points at increasingly large distances (the distance is referred to as the length scale of the clique complex). We denote the number of k-simplices in the complex at length scale i as $|S^i_k|$.

We then compute the Betti numbers β^i_k (the number of k-dimensional holes at a given length scale i) or the persistent Betti numbers $\beta^{i,j}_k$ (the number of k-dimensional holes present at both scale i and scale j) of the simplicial complex. The persistent Betti numbers $\beta^{i,j}_k$ are used to infer the dominant topological features, considered to be those with the longest persistence as the length scale is increased. The births and deaths of features are typically plotted on a "persistence diagram." Different datasets can be compared by using stable distance measures between their diagrams, or by vectorizing the diagrams and using kernel methods or neural networks. For graphs, there is only a single length scale i, and so β^i_k is the quantity of interest. For statements common to both $\beta^{i,j}_k$ and β^i_k, we will use the notation β^*_k. Typical classical applications consider low values of k, motivated primarily by computational cost and interpretability of the resulting topological features.

Dominant resource cost/complexity

Quantum algorithms naturally estimate $\beta^*_k/|S^i_k|$ to additive error ϵ [709, 469, 514, 755, 143]. For a complex built from N data points, we can either use N

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

qubits to encode the simplicial complex, or $O(k\log(N))$ qubits when $k \ll N$. Quantum algorithms have two subroutines:

(i) Preparing a state that encodes the simplices present in the simplicial complex. This reduces to finding k-simplices present in the complex at the given length scale (using either classical rejection sampling or Grover's algorithm / amplitude amplification). Using Grover's algorithm this scales as $\left(\binom{N}{k+1}/|S_k^i|\right)^{1/2}$. More efficient clique-finding methods can be used for special classes of graphs [474].

(ii) Projecting onto the eigenspace of an operator that encodes the topological features of the complex in the amplitude of the quantum state (using either quantum phase estimation or quantum singular value transformation). This introduces a dependence on the gap(s) Λ of the operator(s) used to encode the topology.

The most efficient approaches use amplitude estimation to compute the normalized (persistent) Betti number. The most expensive subroutines within the quantum algorithms are the membership oracles that determine if a given simplex is present in the complex, the cost of which we denote by m_k. In the classically challenging clique-dense regime, the overall cost of the most efficient known quantum algorithms for computing $\beta_k^*/|S_k^i|$ to error ϵ is approximately

$$\widetilde{O}\left(\frac{m_k}{\epsilon}\sqrt{\frac{\beta_k^*}{|S_k^i|}}\left(\sqrt{\frac{\binom{N}{k+1}}{|S_k^i|}}+\frac{\mathrm{poly}(N,k)}{\Lambda}\right)\right).$$

When we choose $\epsilon = \Delta/|S_k^i|$ (i.e., computing β_k^* to constant additive error Δ) the complexity is

$$\widetilde{O}\left(\frac{m_k\sqrt{\beta_k^*}}{\Delta}\left(\sqrt{\binom{N}{k+1}}+\frac{\sqrt{|S_k^i|}\cdot\mathrm{poly}(N,k)}{\Lambda}\right)\right).$$

It is clear that regardless of how $|S_k^i|$ and β_k^* scale, the runtime is polynomial in $\binom{N}{k+1}$.

When we choose $\epsilon = r\beta_k^*/|S_k^i|$ (i.e., computing β_k^* to relative error r) the complexity is

$$\widetilde{O}\left(\frac{m_k}{r}\left(\sqrt{\frac{\binom{N}{k+1}}{\beta_k^*}}+\sqrt{\frac{|S_k^i|}{\beta_k^*}}\cdot\frac{\mathrm{poly}(N,k)}{\Lambda}\right)\right).$$

For clique-dense complexes with large Betti numbers, where β_k^* and $|S_k^i|$ are only polynomially smaller than $\binom{N}{k+1}$, the runtime is polynomial in N and k.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Existing resource estimates

In [755], the gate depth (and non-Clifford gate depth) of all subroutines (including explicit implementations of the membership oracles) was established for computing $\beta_k^{i,j}$ and β_k^i. However, that reference did not consider a final compilation to T/Toffoli gates for concrete problems of interest.

In [143], the Toffoli complexity of estimating β_k^i was determined. The Toffoli complexity for estimating β_k^i to relative error for a family of graphs with large β_k^i was determined for $k = 4, 8, 16, 32$ and $N \leq 10^4$. The resulting Toffoli counts ranged from 10^8 ($N = 100, k = 4$) to 10^{17} ($N = 10^4, k = 32$), using $O(N)$ logical qubits.

Caveats

Quantum algorithms naturally solve a different problem than the textbook classical algorithm solves. Namely, the quantum algorithm estimates $\beta_k^*/|S_k^i|$ to error ϵ, with runtime poly(ϵ^{-1}), for a single length scale (pair of length scales). The algorithm must be repeated for all length scales to compute the persistence diagram. In contrast, the textbook classical algorithm for dimension k and length scale j exactly computes the full persistence diagram for all $\beta_{k' \leq k}^{i,j' \leq j}$.

Quantum algorithms (and classical algorithms based on the power method discussed below) depend on the eigenvalue gap(s) Λ of the operator(s) that encode the topology. The scaling of these gaps has not been studied for typical applications.

Finally, typical classical applications consider dimension $k \leq 3$, and applications of high-dimensional Betti numbers are not yet known.

Comparable classical complexity and challenging instance sizes

While classical algorithms are technically efficient for constant dimension k, they are limited in practice. For a number of benchmark calculations on systems with up to 10^9 simplices, we refer to [819].

The textbook classical algorithm, which for dimension k and length scale j exactly computes the full persistence diagram for all $\beta_{k' \leq k}^{i,j' \leq j}$ (rather than a single Betti number), has a worst-case scaling of $O(|S_{k,k+1}^j|^\omega)$ where $\omega \leq 2.37$ is the exponent of matrix multiplication, and we have defined a shorthand notation $|S_{k,k+1}^j| := |S_k^j| + |S_{k+1}^j|$ [764]. In practice, the textbook classical algorithm is observed to scale as $O(|S_{k,k+1}^j|)$ due to sparsity in the complex [764]. Moreover, for problems that do not naturally have this sparse structure, well-studied classical heuristics can be applied to sparsify the complex [765].

Classical algorithms based on the power method [403] can achieve worst-case scaling of $O(|S_k^i|)$ for computing individual Betti numbers β_k^i (an improve-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ment over the worst-case scaling of the textbook algorithm above). The classical power method scales approximately as

$$\widetilde{O}\left(\frac{|S_k^i|(Nk\beta_k^i + (\beta_k^i)^2)\lambda_{\max}}{\Lambda}\log\left(\frac{1}{\epsilon}\right)\right)$$

to compute β_k^i to additive error ϵ, where $\lambda_{\max}$ is a bound on the largest eigenvalue of the operator encoding the topology. The power method has recently been extended to compute persistent Betti numbers, with a similar complexity [755]. Although the power method for persistent Betti numbers is more efficient than the worst-case performance of the textbook classical algorithm described above, it must be repeated for each pair of length scales to compute the persistence diagram, which is a disadvantage in practice.

Recently, randomized classical algorithms have been proposed for estimating $\beta_k^i/|S_k^i|$ to additive error [143, 53]. The algorithm of [53] scales as

$$\left(\frac{N}{\lambda_{\max}}\right)^{O\left(\frac{1}{\sqrt{\Lambda}}\log\left(\frac{1}{\epsilon}\right)\right)} \cdot \mathrm{poly}(n)$$

assuming that we can efficiently sample and check k-simplices. When $k = \Omega(N)$, the algorithm runs in polynomial time for clique complexes with constant gap Λ and error $\epsilon = \Omega(1/\mathrm{poly}(N))$ (or ϵ constant and $\Lambda = \Omega(1/\log^2(N))$).

Speedup

As discussed above, quantum algorithms naturally compute $\beta_k^*/|S_k^i|$ to additive error ϵ, with runtime $\mathrm{poly}(\epsilon^{-1})$. A number of complexity-theoretic results have been shown for this problem. Reference [474] showed that estimating the kernel dimension of general Hamiltonians is DQC1-hard.[12] In [214], it was shown that estimating normalized quasi-Betti numbers (which accounts for miscounting low-lying but nonzero singular values) of general cohomology groups is also DQC1-hard. The hardness of estimating normalized (persistent) Betti numbers of a clique complex, subject to a gap assumption of $\Lambda = \Omega(1/\mathrm{poly}(N))$—which is the problem solved by existing quantum algorithms—has not been established (see [214, Section 1.1]).

Reference [903] showed that determining if the Betti number of a (clique-dense) clique complex is nonzero is NP-hard in general. This was superseded by the results of [319, 620] which showed that this problem is QMA_1-hard.

We can consider the speedup of quantum algorithms for TDA in two regimes, constant additive error and relative error:

[12] DQC1 is a complexity class that is physically motivated by the "one clean qubit model" [635]. This model has a single pure state qubit which can be initialized, manipulated, and measured freely, as well as $N - 1$ maximally mixed qubits.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- For constant additive error, the most natural comparison is between the quantum algorithm and classical algorithms based on the power method. Quantum algorithms are able to achieve a quadratic speedup over the classical power method in the clique-dense regime [403, 755]. An exponential speedup is not possible for general graphs, due to the aforementioned NP- and QMA-hardness results [903, 319, 620].
- For the task of computing β_k^i to relative error, graph families have been identified for which the quantum algorithm provides superpolynomial [143] or quartic [143, 903] speedups over the classical power method. Recently introduced randomized classical algorithms [143, 53] may scale efficiently for this same task of estimating β_k^i to relative error. For example, when $k = \Omega(N)$ the algorithm of [53] runs in polynomial time for clique complexes with constant gap Λ and error $\epsilon = \Omega(1/\text{poly}(N))$ or ϵ constant and $\Lambda = \Omega(1/\log^2(N))$. These are more restrictive conditions than quantum algorithms (which can simultaneously have both $\Lambda, \epsilon = \Omega(1/\text{poly}(N))$). These features will not occur for all graphs.

NISQ implementations

In [13], a NISQ-amenable compilation of the quantum algorithm described above was proposed, trading deep quantum circuits for many repetitions of shallower circuits, which comes at the cost of worsening the asymptotic scaling of the algorithm (see the table in [755] for a quantitative comparison). A proof-of-principle experiment was performed demonstrating this method [13]. In [214], it was shown that the TDA problem can be mapped to a fermionic Hamiltonian, and it was proposed to use the variational quantum eigensolver to find the ground states of this Hamiltonian (the degeneracy of which gives β_k^i). It is unclear what ansatz circuits one should use to make this approach advantageous compared to classical algorithms, as naive (e.g., random) trial states would have exponentially small overlap with the target states.

Outlook

The complexity-theoretic results that provide evidence for the classical hardness and quantum tractability of estimating normalized (persistent) Betti numbers suggest that quantum algorithms for TDA may be an interesting area to search for new quantum speedups.

Nevertheless, it is important to emphasize that current quantum algorithms do not provide more than quadratic speedups for the practical problem solved in current TDA applications, and complexity-theoretic results suggest that exponential speedups will not be possible for general graphs for this task.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

As such, an important open problem is to identify applications for the task naturally solved by quantum computers (providing relative error estimates for clique-dense graphs with large Betti numbers). If new applications can be identified for datasets that are both clique-dense and have large high-dimensional (persistent) Betti numbers (that are practically interesting to compute to relative error), then quantum algorithms may be of practical relevance.

9.5 Quantum neural networks and quantum kernel methods

Overview

In this section, we discuss two collections of proposals to use a quantum computer as a platform to execute machine learning models, often known as *quantum neural networks* and *quantum kernel methods*. Some early ideas in this space were motivated by the constraints of near-term, "NISQ" [843] devices. Despite this, not all subsequent proposals are necessarily implementable on NISQ devices. Moreover, the proposals need not be restricted to running on NISQ devices, but could also be run on devices with explicit quantum error correction. For simplicity, we present concrete examples based on supervised machine learning tasks. However, outside of these examples, we keep our discussion more general and note that the techniques are also applicable to other settings, such as unsupervised learning and generative modeling.

Given access to some data, our goal is to obtain a function or distribution that emulates certain properties of the data, which we will call a *hypothesis*. This is obtained by first defining a *hypothesis set* or *model family*, and using a learning algorithm to output a hypothesis from this set. For example, in supervised learning, we have data $x_i \in X$ that have respective labels $y_i \in Y$. The goal is then to find a hypothesis function $h : X \to Y$ that approximates the "true" unknown underlying labeling function, such that it correctly labels previously unseen data with high probability. Note that we have left the exact descriptions of the sets X and Y ambiguous. They could, for instance, correspond to sets of numbers or vectors. More generally, this description encompasses the possibility of operating on quantum data such that each x_i or y_i corresponds to a quantum state.

Quantum neural networks and quantum kernel methods use a quantum computer to assist in constructing the model family, in place of a classical model such as a neural network. Specifically, here we prepare some quantum state(s) encoding the data and measure some observable(s) to ultimately construct model predictions. We first elaborate on both quantum neural networks and quantum kernel methods.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Quantum neural networks

Actual end-to-end problem(s) solved: Given data x, we consider a model constructed from a parameterized quantum circuit:

$$h_{\boldsymbol{\theta}}(x) = f(\mathrm{tr}[\rho(x,\boldsymbol{\theta})O]), \tag{9.13}$$

where $\rho(x,\boldsymbol{\theta})$ is a quantum state (output of some parameterized quantum circuit) that encodes both the data x as well as a set of adjustable parameters $\boldsymbol{\theta}$, O is some chosen measurement observable, and f is some function that can be enacted as classical postprocessing on the measurement result (we remark that O itself can also be trainable [475], but we do not explicitly indicate this in the notation for simplicity of exposition). As a basic example, if x corresponds to a classical vector, $\rho(x,\boldsymbol{\theta})$ could correspond to initializing in the $|0\rangle\langle 0|$ state and applying some data-encoding gates $U(x)$ followed by parameterized gates $V(\boldsymbol{\theta})$. Alternatively, the data itself could be a quantum state, and a more general operation in the form of a parameterized channel $\mathcal{V}(\boldsymbol{\theta})$ could be applied. There is also no a priori reason why data encoding and trainable gates need to be applied each once in separate steps rather than in a mixed or repeated fashion.

The model is optimized via a learning algorithm which aims to find the optimal parameters $\boldsymbol{\theta}^*$ by minimizing a loss function. For instance, in supervised learning, given some labeled training dataset $T = \{(x_i, y_i)\}$, a suitable choice of loss should compare how close each $h_{\boldsymbol{\theta}}(x_i)$ is to the true label y_i for all data in T. The quality of the model can then be assessed on a set of previously unseen data outside of T. It is important to pause here and reflect that an optimized loss does not guarantee good performance on unseen data. This is referred to in the literature as the gap between *empirical* and *total* risk, or simply the generalization gap/error. Conversely, a small generalization error alone is not sufficient to guarantee good performance (one should then also ask for a good loss on training data).

We remark that the setting we presented has substantial overlap with the setting of variational quantum algorithms (VQAs)—indeed, a quantum neural network can be thought of as a VQA that incorporates data—thus, the same challenges and considerations that apply to VQAs also apply here. There will additionally be extra considerations due to the role of the data.

Dominant resource cost/complexity: The encoding of data x and parameters $\boldsymbol{\theta}$ in Eq. (9.13) should be sufficiently expressive that it (i) leads to good performance on data and (ii) is (at minimum) not efficiently simulable classically [243], if one is to seek quantum advantage. These requirements set some criteria for minimum circuit complexity.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

The learning algorithm to find optimal parameters is usually performed by classical heuristics, such as gradient descent, and can have significant time overhead, requiring evaluation of Eq. (9.13) at many parameter values (see Chapter 20 on VQAs for more details).

The size of the training dataset required can also have direct implications for runtime, with a larger amount of training data typically taking a longer time to process. Reference [235] proves that good generalization can be achieved with the size of the training data $|T|$ growing in tandem with the number of adjustable parameters M. Specifically, it is shown that the generalization error with high probability scales as $O(\sqrt{M \log(M)/|T|})$. Thus, only a mild amount of data is required for good generalization. We stress again that this alone does not say anything about the ability for quantum neural networks to obtain low training error.

Scope for advantage: Quantum neural networks could achieve advantage in a number of ways, for example, by improving on runtime or by using less training data. In supervised learning settings, generalization performance is a separate consideration and an additional domain for possible advantage. Machine learning with quantum neural networks has yielded some promising performance empirically and encouraging theoretical guarantees exist for certain stages of the full pipeline in restricted settings [901, 234, 235, 700, 1070] (loss minimization can remain a challenge [663, 243], see Chapter 20 on VQAs again for more details). Nevertheless, there are currently no practical use cases with full end-to-end performance guarantees in the same way that we have for other quantum algorithms. However, due to the heuristic nature of classical machine learning, one may debate whether such a guarantee is possible, or even if seeking theoretical quantum advantage in the traditional algorithmic sense is the most appropriate goal [915].

Quantum kernel methods

Actual end-to-end problem(s) solved: Quantum kernel methods are a quantum instance of a class of techniques known as kernel methods, of which support vector machines are a prominent example. We first briefly review the general framework. Given a dataset $T = \{x_i\} \subset X$, the model can be written as

$$h_{\boldsymbol{\alpha}}(x) = \sum_{i\colon x_i \in T} \alpha_i k(x, x_i)\,, \tag{9.14}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots)$ is a vector of parameters to be optimized, and $k(x, x')\colon X \times X \to \mathbb{R}$ is a measure of similarity known as the kernel function. This model has several theoretical motivations:

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- The matrix with entries $K_{ij} = k(x_i, x_j)$ is usually defined to be symmetric positive semidefinite for any choice of $\{x_1, \ldots, x_m\} \subseteq X$ and $k(x_i, x_j)$. By Mercer's theorem, it is thus an inner product of feature vectors $\phi(x_i), \phi(x_j)$ which embed the data x_i and x_j in a (potentially high-dimensional) Hilbert space. Linear statistical methods can be used to learn a linear function in this high-dimensional space, only using the information of the inner products $k(x_i, x_j)$ and never having to explicitly evaluate $\phi(x_i)$ and $\phi(x_j)$, which can be much harder to compute.
- The Representer Theorem [905] states that the optimal model over the dataset T (optimal for T, though not necessarily for expanded datasets) can be expressed as a linear combination of kernel values evaluated over T—that is, the optimal model exactly takes the form in Eq. (9.14). This is known as the kernel trick.
- Further, if the loss function is convex, then the dual optimization program to find the optimal parameters $\boldsymbol{\alpha}^*$ is also convex [913].

A key question that remains is then how to choose a kernel function. Quantum kernel methods embed data in quantum states, and thus evaluate $k(x_i, x_j)$ on a quantum computer. Similar to quantum neural networks or any other quantum model, the quantum kernel should be hard to simulate classically [243]. As an example, we present two common choices of quantum kernel (see [425] for a more general discussion).

- The fidelity quantum kernel

$$k_F(x, x') = \mathrm{tr}[\rho(x)\rho(x')]\,, \tag{9.15}$$

  which can be evaluated either with a SWAP test or, given classical data with unitary embeddings, it can be evaluated with the overlap circuit $|\langle 0|U(x')^\dagger U(x)|0\rangle|^2$.
- The fidelity kernel can run into issues for high-dimensional systems (increasing qubit count), as the inner product in Eq. (9.15) can be very small for $x \neq x'$. This motivated the proposal of a family of projected quantum kernels [542], of which one example is the Gaussian projected quantum kernel

$$k_P(x, x') = \exp\left(-\gamma \sum_{\ell=1}^{n} \left\|\rho_\ell(x) - \rho_\ell(x')\right\|_2^2\right), \tag{9.16}$$

  where $\rho_\ell(x)$ is the reduced density matrix of the n-qubit state $\rho(x)$ on qubit ℓ, and γ is a hyperparameter.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost/complexity: During the optimization of the dual program to find the optimal parameters α^*, $O(|T|^2)$ expectation values corresponding to the kernel values in Eq. (9.14) need to be accurately evaluated, as well as when computing $h_{\alpha^*}(x)$ for a new data point x with the optimized model. This can lead to a significant overhead in applications with large datasets. Alternatively, the primal optimization problem has reduced complexity in the dataset size, but greatly exacerbated dependence on the error [415]. The gate complexity is wholly dependent on the choice of data encoding leading to the kernel function. As the kernel function should be classically nonsimulable, this sets some minimum requirements in terms of circuit complexity. However, in the absence of standardized techniques for data encoding, it is hard to make more precise statements.

Scope for advantage: In [704], the authors demonstrate that using a particular constructed dataset and data embedding, concrete quantum advantage can be obtained for a constructed machine learning problem based on the discrete logarithm problem. The original work was based on the fidelity kernel, but a similar advantage can also be more simply obtained for the projected quantum kernel [542] and adapted beyond kernel methods to the reinforcement learning setting [573]. Beyond this, concrete advantage (up to similar computational assumptions) can be shown more generally for any learning problem where the underlying (unknown, to be learned) labeling function constitutes a BQP-hard family [473]. While great strides have been made in understanding the complexity of quantum kernel methods [88, 542], at present there do not yet exist examples of explicit end-to-end theoretical guarantees of advantage for classical data relevant for a real-world problem. As with quantum neural networks, it may be debated whether or not this is a reasonable question for theoretical research efforts.

Caveats

One consideration we have not discussed so far is how to encode classical data into a quantum circuit, which is a significant aspect of constructing the quantum model. There are many possibilities, such as amplitude encoding or encoding data into rotation angles of single-qubit rotations (see, e.g., [711, 513, 547, 664]). While certain strategies are popular, there is no universal strategy. In general, it is unclear what is the best choice for a given problem at hand, and thus selecting the data-encoding strategy can itself be a heuristic process. The same question extends to the choice of quantum neural network or quantum kernel. While certain choices may perform well in specific problem instances,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

there is at present a lack of strong evidence why such approaches may be advantageous over their classical counterparts in general.

While optimization of parameterized quantum circuits is predominantly a concern for quantum neural networks, the search for good quantum kernels has also motivated proposals of trainable kernels [547, 414, 436] where a parameterized quantum circuit is used to construct the quantum kernel (note that this is distinct from the "classical" optimization of $\boldsymbol{\alpha}$ in Eq. (9.14)). In the case that the parameter optimization process is performed using heuristics, it is subject to the same challenges and considerations that arise with VQAs (see Chapter 20 for more details).

Finite statistics is an important consideration for both settings. Where there is optimization of parameterized quantum circuits, one must take care to avoid the barren plateau phenomenon [663] (again, see Chapter 20 for more details and further references). Analogous effects can also occur in the kernel setting [655], which can arise even purely due to the data-encoding circuit [542, 988].

Outlook

The use of classical machine learning models is generally heuristic, guided by empirical evidence or sometimes physical intuition. Despite this, classical machine learning has found remarkable success in solving many practical problems of interest. The quantum techniques outlined in this section also broadly follow this approach (although theoretical progress has also been substantial in certain areas), and there is no a priori reason why they cannot also be useful. Nevertheless, it can be challenging to make concrete predictions for quantum advantage, particularly for learning problems with classical data (see [704, 473] as some exceptions). For practical problems this is exacerbated by our limited analytic understanding for end-to-end applications, even in the fully classical setting. Indeed, it may ultimately be challenging to have the same complete end-to-end theoretical analysis that other quantum algorithms enjoy, aside from a few select examples [915]. Within the realm of quantum data, there appears to be ripe potential for concrete provable advantage [543, 265, 236], however, this is beyond the scope of this section.

Further reading

We refer the reader to [913, 547] for pedagogical expositions of quantum kernel methods, to [118, 241] for comprehensive reviews of quantum neural networks, and to [242] for a review of quantum machine learning models at large, including an exposition of machine learning with quantum data.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

PART II

QUANTUM ALGORITHMIC PRIMITIVES

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Overview

To deliver an advantage over classical approaches, end-to-end quantum solutions must exploit known quantum phenomena capable of providing a quantum speedup. The disparate collection of known quantum applications is built from a common group of *quantum algorithmic primitives*, which are the source of quantum advantage. Algorithmic primitives are typically not suited for directly solving an end-to-end problem, due to their reliance on unspecified oracles or because their input and/or output does not exactly match that of the end-to-end problem (e.g., some primitives output a quantum state rather than classical data, and thus they have no direct classical analog). Nevertheless, it can be very fruitful to think of algorithms as compositions of different algorithmic primitives, both for higher-level intuitive overview and for independently studying and optimizing the primitives themselves.

This part surveys a variety of quantum algorithmic primitives. For each, we sketch the basic idea of what they do and how they work, as well as discussing example use cases and important caveats. We generally assume that these primitives will need to be implemented in fault-tolerant fashion when they are used within an end-to-end solution for a given application, but we comment on NISQ implementations in passing.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

10

Quantum linear algebra

At a high level of abstraction, quantum computers compose unitary matrices, and do so with classically unparalleled efficiency. This hints at quantum speedups for linear algebra tasks. However, often one needs to work with large non-unitary matrices; thus, for performing general linear algebra tasks we often wish to embed certain non-unitary matrices into unitary matrices represented by efficient quantum circuits, and then apply them to quantum states, take their sums or products, or implement more general matrix functions. These tasks are collectively referred to as "quantum linear algebra," the building blocks of which are discussed in this chapter.

The techniques described in this chapter evolved over the past decades and converged to the presented unified framework within several distinct research threads. Block-encodings emerged as a natural approach for embedding non-unitary matrices into quantum circuits, inspired by approaches based on purification, dilation (e.g., Stinespring representation [1050] or Stinespring dilation [1044]), and postselection. Quantum signal processing (QSP) was discovered as a byproduct of the characterization of simple single-qubit pulse sequences used in nuclear magnetic resonance [719], for synthesizing polynomial transformations applicable to a "signal parameter" encoded as a matrix element of a single-qubit rotation matrix. Meanwhile, it was extensively studied how matrix functions could be synthesized using the linear combinations of unitaries technique on matrix exponentials implemented by Hamiltonian simulation [281, 48, 248], or Chebyshev polynomials of operators implemented via quantum walk techniques [135, 136, 282]. Such matrix exponentials or Chebyshev polynomials can be implemented, for example, via qubitization of a block-encoded operator. In parallel to progress on advanced amplitude amplification [466, 1067] techniques, it was recognized [716, 717] that QSP can be "lifted" for applying polynomial transformations to the eigenvalues of quantum walk operators (such as those implemented by qubitization), and thus for

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

implementing a rich family of matrix functions, immediately yielding an optimal algorithm for time-independent Hamiltonian simulation. The concepts of qubitization and QSP were later generalized and unified into the framework of quantum singular value transformation [431], providing generalizations and more efficient implementations of a number of existing quantum algorithms and leading to the discovery of several new algorithms.

The authors are grateful to Lin Lin for reviewing this chapter.

10.1 Block-encodings

Rough overview (in words)

In a quantum algorithm, the quantum gates that are applied to quantum states are necessarily unitary operators. However, one often needs to apply a linear transformation to some encoded data that is not represented by a unitary operator, and furthermore, one generally needs coherent access to these non-unitary transformations. How can we encode such a non-unitary transformation within a unitary operator? Block-encoding is one method of providing exactly this kind of coherent access to generic linear operators. Block-encoding works by embedding the desired linear operator as a suitably normalized block within a larger unitary matrix, such that the full encoding is a unitary operator, and the desired linear operator is given by restricting the unitary to an easily recognizable subspace. To be useful for quantum algorithms, this block-encoding unitary must also be realized by some specific quantum circuit acting on the main register and additional ancilla qubits.

Block-encodings are ubiquitous within quantum algorithms, but they have both benefits and drawbacks. They are easy to work with, since one can efficiently perform manipulations of block-encodings, such as taking products or convex combinations. On the other hand, this improved working efficiency comes at the cost of having more limited access. For example, if a matrix is stored in classical random access memory, the matrix entries can be explicitly accessed with a single query to the memory, whereas if one only has access to a block-encoding of the matrix, estimating a matrix entry to precision ε requires $O(1/\varepsilon)$ uses of the block-encoding unitary in general (by utilizing an amplitude estimation subroutine).

Block-encodings also provide a layer of abstraction that assists in the design and analysis of quantum algorithms. One can simply assume access to a block-encoding and count the number times it is applied. To run the algorithm, it is

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

necessary to choose a method for implementing the block-encoding. There are many ways of constructing block-encodings that could be suited to the structure of the input. For instance, there are efficient block-encoding strategies for density matrices, positive operator-valued measures (POVMs), Gram matrices, sparse matrices, matrices that are stored in quantum data structures, structured matrices, and operators given as a linear combination of unitaries (with a known implementation). We discuss these constructions below. For unstructured, dense matrices, the strategy for Gram matrices can be instantiated using state preparation and quantum random access memory (QRAM) as subroutines. For more details on a particular block-encoding scheme for loading matrices of classical data, see Section 17.3 on block-encoding dense matrices of classical data.

Rough overview (in math)

Our goal is to build a unitary operator that gives coherent access to an $M \times M$ matrix A (we will later relax the assumption that A is square), with normalization $\alpha \geq \|A\|$, where $\|A\|$ denotes the spectral norm of A. As the name suggests, block-encoding is a way of encoding the matrix A as a block in a larger unitary matrix

$$U_A = \begin{array}{c} \\ |0^a\rangle \\ |0^a\rangle_\perp \end{array} \begin{array}{c} \begin{array}{cc} |0^a\rangle & |0^a\rangle_\perp \end{array} \\ \begin{pmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix} \end{array},$$

where the labels $|0^a\rangle$ and $|0^a\rangle_\perp$ indicate which portion of the vector space each block corresponds to—specifically, whether the first a qubits are equal to or orthogonal to the state $|0^a\rangle$, respectively. Three of the four blocks are unspecified and can take on any values such that U_A is unitary. More precisely, we say that the unitary U_A is an (α, a, ϵ)-block-encoding of the matrix $A \in \mathbb{C}^{M \times M}$ if

$$\|A - \alpha(\langle 0^a| \otimes I) U_A (|0^a\rangle \otimes I)\| \leq \epsilon, \tag{10.1}$$

where $a \in \mathbb{N}$ is the number of ancilla qubits used for embedding the block-encoded operator, and $\alpha, \epsilon \in \mathbb{R}_+$ define the normalization and error, respectively. Note that $\alpha \geq \|A\| - \epsilon$ is necessary for U_A to be unitary. The definition above can be extended for general matrices, though additional embedding or padding may be needed (e.g., to make the matrix square).

Once a block-encoding is constructed, it can be used in a quantum algorithm to apply the matrix A to a quantum state by applying the unitary U_A to the larger quantum system. The application of the block-encoding can be thought of as a probabilistic application of A—applying U_A to $|0^a\rangle|\psi\rangle$ and postselecting on the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

first register being in the state $|0^a\rangle$ gives an output state proportional to $A|\psi\rangle$ in the second register.

There are several ways of implementing block-encodings based on the choice of matrix A [431, Section 4.2].[1]

- Unitary matrices are $(1, 0, 0)$-block-encodings of themselves. Controlled unitaries (e.g., CNOT) are essentially $(1, 1, 0)$-block-encodings of the controlled operation.
- Given an s-qubit density matrix ρ and an $(a+s)$-qubit unitary G that prepares a *purification* of ρ as $G|0^a\rangle|0^s\rangle = |\rho\rangle$ (s.t. $\mathrm{tr}_a|\rho\rangle\langle\rho| = \rho$, where tr_a denotes trace over the first register), then the operator [717]

$$(G^\dagger \otimes I_s)(I_a \otimes \mathrm{SWAP}_s)(G \otimes I_s)$$

  is a $(1, a + s, 0)$-block-encoding of the density matrix ρ, where I_x denotes the identity operator on a register with x qubits, and SWAP_s denotes the operation that swaps two s-qubit registers [431, Lemma 45].
- Similarly, one can construct block-encodings of POVM operators, given access to a unitary that implements the POVM [45]. Specifically, if U is a unitary that implements the POVM M to precision ϵ, such that for all s-qubit density operators ρ we have

$$\left|\mathrm{tr}(\rho M) - \mathrm{tr}\Big[U(|0\rangle\langle 0|^{\otimes a} \otimes \rho)U^\dagger(|0\rangle\langle 0| \otimes I_{a+s-1}))\Big]\right| \leq \epsilon,$$

  then $(I_1 \otimes U^\dagger)(\mathrm{CNOT} \otimes I_{a+s-1})(I_1 \otimes U)$ is a $(1, 1 + a, \epsilon)$-block-encoding of M [431, Lemma 46].
- One can also implement a block-encoding of a Gram matrix using a pair of state preparation unitaries U_L and U_R. In particular, the product

$$U_A = U_L^\dagger U_R$$

  is a $(1, a, 0)$-block-encoding of the Gram matrix A whose entries are $A_{ij} = \langle\psi_i|\phi_j\rangle$, where [431, Lemma 47]

$$U_L|0^a\rangle|i\rangle = |\psi_i\rangle, \qquad U_R|0^a\rangle|j\rangle = |\phi_j\rangle.$$

- One can generalize the above strategy from Gram matrices to arbitrary matrices to produce (α, a, ϵ)-block-encodings of general matrices A, where again $\alpha \geq \|A\|$. See Section 17.3 on block-encoding dense matrices of classical data for details.
- Sparse matrices: Given a matrix $A \in \mathbb{C}^{2^w \times 2^w}$ that is s_r-row sparse and s_c-column sparse (meaning each row and column has at most s_r and s_c nonzero

[1] References to locations in [431] typically refer to the longer arXiv version, rather than the STOC version.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

entries, respectively), then, defining $\|A\|_{\max} = \max_{i,j} |A_{ij}|$, one can create a $(\sqrt{s_r s_c}\|A\|_{\max}, w+3, \epsilon)$-block-encoding of A using oracles O_r, O_c, and O_A, defined as follows [431, Lemma 48]

$$\begin{aligned} O_r &: |i\rangle|k\rangle \mapsto |i\rangle|r_{ik}\rangle, && \forall i \in [2^w]-1, k \in [s_r], \\ O_c &: |\ell\rangle|j\rangle \mapsto |c_{\ell j}\rangle|j\rangle, && \forall \ell \in [s_c], j \in [2^w]-1, \\ O_A &: |i\rangle|j\rangle|0^b\rangle \mapsto |i\rangle|j\rangle|A_{ij}\rangle, && \forall i, j \in [2^w]-1. \end{aligned}$$

In the above, r_{ij} is the index of the j-th nonzero entry in the i-th row of A (or $j + 2^w$ if there are less than i nonzero entries), c_{ij} is the index of the i-th nonzero entry in the j-th column of A (or $i + 2^w$ if there are less than j nonzero entries), and $|A_{ij}\rangle$ is a b-bit binary encoding of the matrix element A_{ij}. To build the block-encoding, one needs one query to each of O_r and O_c, and two queries of O_A. This input model is known as the sparse access model. If, in addition to being sparse, the matrix also enjoys some additional *structure*, for example, there are only a few distinct values that the matrix elements can take, the complexity can be further improved [969, 227]. Finally, note that the sparsity dependence can be essentially quadratically improved—reducing the block-encoding normalization factor from $\sqrt{s_r s_c}\|A\|_{\max}$ to $(\max(s_r, s_c))^{(1+o(1))/2}\|A\|_{1\to 2}$, where $\|A\|_{1\to 2} = \max_v \|Av\|_2/\|v\|_1$—using advanced Hamiltonian simulation techniques [714, Theorem 2] combined with taking the logarithm of unitaries [431, Corollary 71], however, the resulting subroutine may be impractical and comes with a worse precision dependence.

- For matrices given as a linear combination of unitary operators (LCU), we can block-encode the matrix using the LCU technique [281]. We provide a full description in §Linear combinations of Section 10.2, and only give a brief outline here. For $A = \sum_{i=1}^{L} c_i V_i$ with V_i unitary, we define the oracles PREPARE (acting on $\lceil \log_2(L) \rceil$ ancilla qubits) and SELECT (acting on the ancilla and register qubits), and implement a $(\sum_i |c_i|, \lceil \log_2(L) \rceil, 0)$-block-encoding of A, using $U := \text{PREPARE}^\dagger \cdot \text{SELECT} \cdot \text{PREPARE}$. The Hamiltonians of physical systems can often be written as a linear combination of a moderate number of Pauli operators, leading to a prevalence of this technique in quantum algorithms for chemistry [75, 140] and condensed matter physics [75, 283, 1011].

In addition to the definition of block-encoding in Eq. (10.1), one can also define an asymmetric version as follows

$$\left\| A - \alpha(\langle 0^a| \otimes I) U_A (|0^b\rangle \otimes I) \right\| \leq \epsilon,$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

where a may not equal b. In this case, U_A can be considered to be an $(\alpha, (a, b), \epsilon)$- or an $(\alpha, \max(a, b), \epsilon)$-block-encoding of A. This can be useful for block-encoding a non-square matrix.

Dominant resource cost (gates/qubits)

The complexity of block-encoding an operator depends on the type of data or operator being encoded and any underlying assumptions. For instance, unitaries are naturally block-encodings of themselves, and hence their resource requirements depend entirely on their circuit-level implementation without any additional overhead for being a "block-encoding." By contrast, approaches that make use of state preparation and QRAM to implement the block-encoding tend to have larger complexities, as those two subroutines typically dominate the resource requirements. For example, the best known circuits that implement block-encodings of matrices of classical data for general, dense $N \times N$ matrices use $O(N \log(1/\epsilon))$ qubits to achieve minimum T-gate count (which also scales as $O(N \log(1/\epsilon))$), or a larger $O(N^2)$ number of qubits to achieve minimum T-gate depth (which scales as $O(\log(N) + \log(1/\epsilon))$ [296]. In the sparse access model, one can use $O(w + \log^{2.5}(s_r s_c/\epsilon))$ one- and two-qubit gates, and $O(b + \log^{2.5}(s_r s_c/\epsilon))$ ancilla qubits [431], in addition to the calls to the matrix entry O_A and sparse access oracles O_r and O_c, which must be implemented either by computing matrix entries "on-the-fly" or by using a primitive (see [1084] for asymptotic resource statements for general sparse matrices with varying ancilla availability). Assuming appropriate binary representations of the numbers A_{ij}, the exponents of the above logarithms can be reduced to 1 using the techniques of [894] (see also [75, Section III.D] and [212, Supplementary Material VII.A.2]).

The value of block-encodings is not that it is always cheap to implement them (as it depends on the relevant cost metric and the data access model); rather, the concept of block-encodings is powerful because it allows a practitioner of quantum algorithms to study and optimize the block-encoding construction independently of how it is used within the larger algorithm.

Caveats

The definition of block-encoding requires $\|A\|/\alpha \leq 1$. If $\|A\|/\alpha = 1$, then the block-encoding achieves an optimal normalization factor α. However, note that often the above constructions lead to suboptimal normalization factors in the sense that $\alpha \gg \|A\|$. In practical applications, this suboptimality usually leads to a corresponding increase in the overall complexity.

For a given desired block-encoding, there can be several independent, yet equally valid implementations, and one can sometimes optimize for various re-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

sources when building the block-encoding. For example, many block-encoding strategies require a step in which some classical data is loaded into QRAM, but there are several ways of performing this data-loading step.

When using a block-encoding as part of a larger quantum algorithm, it is important to ensure that the overhead introduced by implementing a block-encoding will not outweigh any potential quantum speedups, as block-encoding can be very resource intensive.

The use of $|0\rangle^{\otimes a}$ as the "signal" state is just one convention—we can use any "signal" state, given a unitary to prepare it [717]. One can also consider a more general definition known as "projected unitary encodings" which allows using an arbitrary subspace, rather than just a state-indexed block [431].

Example use cases

Block-encodings are ubiquitous in quantum algorithms, and they prevail in quantum algorithms that need coherent access to some linear operator or a method of implementing a non-unitary transformation on quantum data. Some specific examples:

- We can manipulate block-encoded operators—for example, take convex or linear combinations, products, tensor products, and other transformations of an input operator.
- The combination of qubitization with quantum signal processing, or quantum singular value transformation can be used to realize algorithms by applying polynomial transformations to block-encoded matrices. Prominent examples are Hamiltonian simulation via qubitization, and matrix (pseudo) inversion [431, Theorem 41] that can be used for solving large linear systems of equations [500] or more generally least-squares regression problems [248].
- Block-encoding can be used to provide coherent access to classical data in a quantum algorithm; for example, loading classical data into a quantum interior point method for portfolio optimization [328].

Further reading

Reference [248] provides an instructive overview of the concept of block-encoding and showcases its power in several applications related to (generalized) regression problems. Meanwhile, [431] is a comprehensive collection of technical results about block-encodings and quantum linear algebra more generally.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

10.2 Manipulating block-encodings

Rough overview (in words)

Given one or more block-encodings, we often want to form a single block-encoding of a product, tensor product, or linear combination of the individual block-encoded operators. This can be achieved as outlined below, using additional ancilla qubits.

Rough overview (in math) and resource cost

We will consider the case of two operators A and B, with straightforward generalizations to additional operators [431]. We are given an (α, a, ϵ_a)-block-encoding U_A of A, and a (β, b, ϵ_b)-block-encoding U_B of B. Operators A and B act on system qubits s.

Products: The operation $U_{AB} := (I_b \otimes U_A)(U_B \otimes I_a)$ is an $(\alpha\beta, a+b, \alpha\epsilon_b + \beta\epsilon_a)$-block-encoding of AB [431, Lemma 53], where I_x denotes the identity operator on x qubits (see Fig. 10.1). For example, if $a = b$, this construction uses twice as many ancilla qubits for block-encoding the product compared to the block-encoding of the individual matrices. In fact, we can assume without loss of generality that $a = b$ (by taking the maximum of the two) and improve the construction using the circuit in Fig. 10.2, which uses $a + 1$ ancilla qubits instead of $2a$. This idea has been generalized to encompass products of L block-encodings using only $a + \lceil \log_2(L) \rceil + 1$ ancillas (rather than aL), where it is known as the "compression gadget"; see [718, Lemma 13] and [379, Lemma 3].

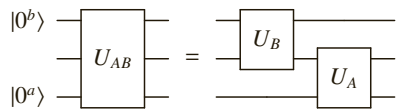


Figure 10.1 Implementing the block-encoding U_{AB} of AB that acts on s qubits. The cost is $a + b$ ancilla qubits, and one call to each of U_A, U_B.

Tensor products: The operation $U_{A\otimes B} := (U_A \otimes U_B)$ is an $(\alpha\beta, a+b, \alpha\epsilon_b + \beta\epsilon_a)$-block-encoding of the operator $A \otimes B$, as depicted in Fig. 10.3.

Linear combinations: Linear combinations of block-encodings can be viewed as a generalization of the linear combination of unitaries (LCU) trick [281]. We wish to implement a block-encoding of $\sum_{i=0}^{L-1} c_i A_i$, where

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

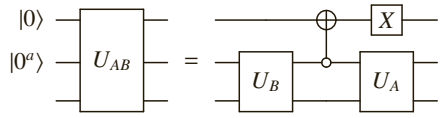


Figure 10.2 Implementing the block-encoding U_{AB} of AB for the case where both U_A and U_B act on a ancilla qubits. The controlled gate is an a-controlled generalized Toffoli gate.

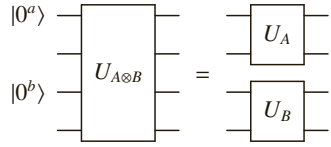


Figure 10.3 Implementing the block-encoding $U_{A\otimes B}$ of $A \otimes B$ that acts on $2s$ qubits. The cost is $a + b$ ancilla qubits, and one call to each of U_A, U_B.

$c_i \in \mathbb{R}$ (the LCU trick can also be extended to complex coefficients) and define $\lambda := \sum_{i=0}^{L-1} |c_i|$. We consider L block-encodings U_i that are $(1, m, \epsilon_i)$-block-encodings of A_i. We note that in cases where the block-encodings have different α_i or m_i values, the former can be absorbed into the c_i values and the latter can be taken as $m = \max_i m_i$.

We first define an operator PREPARE by the following action on $|0^{\lceil \log_2(L) \rceil}\rangle$

$$\text{PREPARE}|0^{\lceil \log_2(L) \rceil}\rangle = \frac{1}{\sqrt{\lambda}} \sum_j \sqrt{|c_j|}|j\rangle$$

that prepares a weighted superposition on an ancilla register, such that the amplitudes are proportional to the square roots of the absolute values of the desired coefficients. We also define[2]

$$\text{SELECT} = \sum_{j=0}^{L-1} |j\rangle\langle j| \otimes \text{sign}(c_j) U_j.$$

We then have the following result

$$\begin{aligned} &\left(\langle 0^{\lceil \log_2(L) \rceil}| \otimes I\right) \text{PREPARE}^\dagger \cdot \text{SELECT} \cdot \text{PREPARE}\left(|0^{\lceil \log_2(L) \rceil}\rangle \otimes I\right) \\ &= \frac{1}{\lambda} \sum_{i=0}^{L-1} c_i U_i \,, \end{aligned} \tag{10.2}$$

[2] To be precise, for $j \notin \{0, 1, \ldots, L-1\}$ we define $\text{sign}(c_j) U_j := I$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

that is, $U_{\mathrm{LC}} := \mathrm{PREPARE}^\dagger \cdot \mathrm{SELECT} \cdot \mathrm{PREPARE}$ is a $(\lambda, \lceil \log_2(L) \rceil, 0)$-block-encoding of the LCU $\sum_i c_i U_i$, as depicted in Fig. 10.4. This is the standard LCU trick [281], and it does not require U_i to be block-encodings (or we can view them as $(1, 0, 0)$-block-encodings of themselves). This technique can be used in Hamiltonian simulation, or to instantiate a block-encoding.

If, as specified above, U_i are block-encodings of $\tilde{A}_i$ (which approximate A_i), we also have the following result

$$\left\| \left(\sum_{i=0}^{L-1} c_i A_i \right) - \lambda \left(\langle 0^{m+\lceil \log_2(L) \rceil}| \otimes I \right) U_{\mathrm{LC}} \left(|0^{m+\lceil \log_2(L) \rceil}\rangle \otimes I \right) \right\| \leq \sum_{i=0}^{L-1} |c_i| \epsilon_i.$$

Hence, U_{LC} is a $(\lambda, \lceil \log_2(L) \rceil + m, \lambda \max_i \epsilon_i)$-block-encoding of $\sum_{i=0}^{L-1} c_i A_i$.

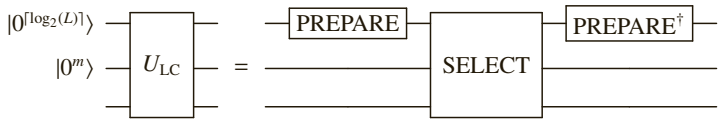


Figure 10.4 Implementing the block-encoding U_{LC} of $\sum_i c_i A_i$ that acts on s qubits. We require $\lceil \log_2(L) \rceil + m$ ancilla qubits. The regular LCU circuit is obtained by omitting the register $|0^m\rangle$ and the requirement that U_i are block-encodings. The gate complexity of PREPARE depends on the coefficients c_i but is $\Theta(L)$ in the worst case (using no additional ancilla qubits) [835]. We can also define PREPARE that leads to entanglement with a garbage register $\mathrm{PREPARE}|0^{\lceil \log_2(L) \rceil}\rangle|0^g\rangle = \lambda^{-0.5} \sum_i \sqrt{|c_i|}|i\rangle|G_i\rangle$, which can be seen to satisfy the relations required to implement the linear combination, Eq. (10.2). It can sometimes (e.g., [75]) be cheaper to implement this garbage-entangled PREPARE; see Section 17.2 on preparing states from classical data. The cost of SELECT depends on the form of U_i, but in the worst case requires $\Theta(L)$ primitive gates and $\Theta(L)$ calls to $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes U_i$ [283, 75], although this can be improved in some relevant special cases (e.g., [1011]). When U_i are multiqubit Pauli operators, [1084] provides a depth-optimized implementation of SELECT that achieves $O(\log(Ln))$ depth using $\Theta(Ln)$ total gates and total ancilla qubits.

Caveats

Performing linear algebraic manipulations of block-encodings using these primitives can quickly increase the ancilla count of the algorithm and worsen the normalization factor of the block-encoding. Amplifying a subnormalized block-encoding is possible, but costly, requiring an amount of time scaling roughly linearly in the amplification factor; see [715, 431]. Given a single block-encoded operator A, the above primitives can be used to implement a block-encoding of a polynomial in A. However, this can be achieved with much lower overhead using quantum singular value transformation (QSVT).

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Example use cases

- Linear combination of block-encodings are used to obtain mixed-parity functions in QSVT required for Hamiltonian simulation.
- LCU trick is used for Hamiltonian simulation via Taylor series and to instantiate block-encodings of chemistry or condensed matter physics Hamiltonians (see, e.g., [75, 1011]).

Further reading

- References [428, Section 3.3] and [687, Section 7.3] contain a comprehensive discussion of manipulating block-encodings, including proofs of many of the results stated above.

10.3 Quantum signal processing

Rough overview (in words)

Quantum signal processing (QSP) [719] describes a method for nonlinear transformations of a signal parameter encoded in a single-qubit gate, using a structured sequence that interleaves the "signal gate" with fixed parameterized "modulation" gates. The technique was originally motivated by the desire to characterize pulse sequences used in nuclear magnetic resonance [719]. Remarkably, it has been shown [719, 477] that there is a rich family of polynomial transformations that are in one-to-one correspondence with appropriate modulation sequences; moreover, given such a polynomial, one can efficiently compute the corresponding modulation parameters.

Even more remarkably, this analysis holds not just for single-qubit "signal gates" but can be extended for multiqubit operators that *act* like single-qubit rotations when restricted to appropriate 2D subspaces [716]. This insight enables the implementation of block-encodings of polynomials of Hermitian or normal matrices when used in conjunction with qubitization. The two-step process of qubitization and QSP can be unified and generalized through quantum singular value transformation (QSVT).

Rough overview (in math)

We follow the "Wx convention" of QSP [431, 744]. We define the single-qubit signal operator

$$W(x) := \begin{pmatrix} x & \mathrm{i}\sqrt{1-x^2} \\ \mathrm{i}\sqrt{1-x^2} & x \end{pmatrix} = \mathrm{e}^{\mathrm{i}\arccos(x)X}$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

which is a single-qubit X rotation. We can verify that

$$W(x)^2 = \begin{pmatrix} 2x^2 - 1 & \cdot \\ \cdot & \cdot \end{pmatrix}$$
$$W(x)^3 = \begin{pmatrix} 4x^3 - 3x & \cdot \\ \cdot & \cdot \end{pmatrix}$$
$$\vdots$$
$$W(x)^n = \begin{pmatrix} T_n(x) & \cdot \\ \cdot & \cdot \end{pmatrix},$$

where $T_n(x)$ is the n-th Chebyshev polynomial of the first kind, showcasing that even a simple sequence of the signal unitaries can implement a rich family of polynomials of the signal x.

More complex behavior is obtained by interleaving $W(x)$ with parameterized single-qubit Z rotations $e^{i\phi_j Z}$. We define a QSP sequence as

$$U_{\mathrm{QSP}}(\Phi) := e^{i\phi_0 Z} \prod_{j=1}^{d} W(x) e^{i\phi_j Z},$$

where Φ denotes the vector of angles $(\phi_0, \phi_1, \ldots, \phi_d)$. The QSP sequence implements the following unitary

$$U_{\mathrm{QSP}}(\Phi) = \begin{pmatrix} P(x) & iQ(x)\sqrt{1-x^2} \\ iQ^*(x)\sqrt{1-x^2} & P^*(x) \end{pmatrix}, \tag{10.3}$$

where $P(x), Q(x)$ are complex polynomials obeying a number of constraints (see below), and $P^*(x)$, $Q^*(x)$ denote their complex conjugates. Note also that the relationship between the sequence Φ and the corresponding polynomial $P(x)$ can be understood through nonlinear Fourier analysis [19].

Dominant resource cost (gates/qubits)

A QSP circuit that implements a degree-d polynomial in the signal parameter requires d uses of $W(x)$ and $d + 1$ fixed-angle Z rotations. There are efficient classical algorithms to determine the angles for a given target polynomial, either using high-precision arithmetic with $\sim d \log(d)$ bits of precision [477] (or more [431]—though this can be mitigated using heuristic techniques [255, 1065]) or in some regimes using more efficient optimization-based or iterative algorithms [356, 1018, 359, 360, 19, 18]. In particular, this line of work has culminated in [18] with an algorithm for finding the angles that is provably numerically stable, that is, it requires only polylog(d/ϵ) bits of precision to achieve ϵ error. An alternative method computes the angles directly

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

if both $P(x)$ and $Q(x)$ are known, and offers a way to compute $Q(x)$ if only $P(x)$ is known [782, 130]. Although these procedures are efficient in theory, in practice it may still be nontrivial to find the angles. Nevertheless, researchers reportedly computed angle sequences corresponding to various degree $d \approx 10^7$ polynomials [782, 130].

Caveats

As discussed above, not all polynomials can be implemented by a QSP sequence. Implementable polynomials must obey a number of constraints, which can be somewhat restrictive. For the standard QSP circuit $U_{\mathrm{QSP}}(\Phi)$ given above, the achievable polynomials pairs $P(x), Q(x) \in \mathbb{C}$ can be characterized by the following three conditions

- $\mathrm{Deg}(P) \leq d$ and $\mathrm{Deg}(Q) \leq d-1$,
- $\mathrm{Parity}(P) = \mathrm{Parity}(d)$ and $\mathrm{Parity}(Q) = \mathrm{Parity}(d-1)$,
- $\forall\ x \in [-1,1] : |P(x)|^2 + (1-x^2)|Q(x)|^2 = 1$ (required for Eq. (10.3) to be unitary).

This last requirement can be particularly limiting. A useful way to circumvent this for real functions is to encode the polynomial in the matrix element $\langle +|U_{\mathrm{QSP}}(\Phi)|+\rangle$ rather than in $\langle 0|U_{\mathrm{QSP}}(\Phi)|0\rangle$, where $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$. This matrix element evaluates to

$$\langle +|U_{\mathrm{QSP}}(\Phi)|+\rangle = \mathrm{Re}[P(x)] + i\sqrt{1-x^2}\,\mathrm{Re}[Q(x)]\,.$$

Given a real target polynomial $f(x)$ with parity equal to $\mathrm{Parity}(d)$, we can guarantee that the matrix element evaluates to $f(x)$ by choosing $\mathrm{Re}[P(x)] = f(x)$ and $\mathrm{Re}[Q(x)] = 0$. The third condition above then reduces to $1 - f(x)^2 = |\mathrm{Im}[P(x)]|^2 + (1-x^2)|\mathrm{Im}[Q(x)]|^2$. By [431, Lemma 6], there exist choices for $\mathrm{Im}[P(x)]$ and $\mathrm{Im}[Q(x)]$ that satisfy this identity as well as the first two conditions above, provided $|f(x)| \leq 1\ \forall\ x \in [-1,1]$. In summary, we may implement any real polynomial $f(x)$ satisfying the requirements [431, Corollary 10]:

- $\mathrm{Deg}(f) = d$,
- $\mathrm{Parity}(f) = \mathrm{Parity}(d)$,
- $\forall\ x \in [-1,1] : |f(x)| \leq 1$.

There are several related conventions considered in the literature for the explicit form of the single-qubit operators used in QSP; a thorough discussion is given in [744, Appendix A]. One common form that links closely to qubitization and QSVT is the reflection convention, which replaces $W(x)$ by the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

reflection

$$R(x) = \begin{pmatrix} x & \sqrt{1-x^2} \\ \sqrt{1-x^2} & -x \end{pmatrix}, \tag{10.4}$$

and adjusts the parameters $\{\phi_j\}$ accordingly [431].

Example use cases

- Functions of Hermitian or normal matrices, in conjunction with qubitization, including for Hamiltonian simulation.
- Functions of general matrices via QSVT.
- Reference [357] applied QSP to beyond-Heisenberg-limit calibration of two-qubit gates in a superconducting system.

Further reading

- A pedagogical discussion of QSP [744].
- Detailed proofs of the key results of QSP [719, 431].
- Lecture notes on QSP [687, Section 7.6].

10.4 Qubitization

Rough overview (in words)

Qubitization is a method for using a block-encoding U_A of a Hermitian operator A to manipulate A, for example, implement A^2, or, more generally, some function $f(A)$ [717]. However, the eigenvalues of U_A are typically unrelated to those of A, and plain repeated applications of U_A do not in general produce the desired behavior. Qubitization converts the block-encoding U_A into a unitary operator W (sometimes called a qubiterate or a qubitized quantum walk operator) having the following guaranteed advantageous properties:

- W is a block-encoding of the operator A.
- The spectrum of W is directly related to the spectrum of A.
- Repeated application of W leads to structured behavior that can be cleanly analyzed.

This combination of features means that qubitization can be used for applying polynomial transformations to the spectrum of A. For example, repeated application of W implements Chebyshev polynomials of A, while other polynomials can also be implemented by using quantum signal processing [716, 717, 431].

The key observation is that a qubitization unitary W has eigenvalues and eigenvectors that relate in a nice way to those of A. Thus, one can also perform

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

quantum phase estimation on W to learn these quantities [841, 139], providing a potentially cheaper alternative to such tasks compared to approaches based on explicit Hamiltonian simulation for implementing $U = \mathrm{e}^{\mathrm{i}At}$.

Rough overview (in math)

We are given a $(1, m, 0)$-block-encoding U_A of Hermitian A, such that

$$A = (\langle 0^m| \otimes I)U_A(|0^m\rangle \otimes I) \iff U_A = \begin{pmatrix} A & \cdot \\ \cdot & \cdot \end{pmatrix}.$$

First, we will assume U_A is also Hermitian (implying $U_A^2 = I$, where I is the identity matrix). Let A have spectral decomposition $A = \sum_\lambda \lambda|\lambda\rangle\langle\lambda|$. An application of U_A to an eigenstate $|\lambda\rangle$ of A gives

$$U_A|0^m\rangle|\lambda\rangle = \lambda|0^m\rangle|\lambda\rangle + \sqrt{1-\lambda^2}|\perp_{0^m,\lambda}\rangle,$$

where $|\perp_{0^m,\lambda}\rangle$ is a state perpendicular to $|0^m\rangle$.[3] Noting $U_A^2 = I$ reveals that the 2D subspace S_λ spanned by $\{|0^m\rangle|\lambda\rangle, |\perp_{0^m,\lambda}\rangle\}$ is invariant under the action of U_A. U_A restricted onto S_λ can be described by the matrix

$$\begin{array}{c} \\ |0^m\rangle|\lambda\rangle \\ |\perp_{0^m,\lambda}\rangle \end{array} \begin{array}{c} \begin{array}{cc} |0^m\rangle|\lambda\rangle & |\perp_{0^m,\lambda}\rangle \end{array} \\ \begin{pmatrix} \lambda & \sqrt{1-\lambda^2} \\ \sqrt{1-\lambda^2} & -\lambda \end{pmatrix} \end{array},$$

which is a 2D reflection with eigenvalues ± 1. Clearly, repeated application of (self-inverse) U_A can have limited effect on any input state. Qubitization uses a reflection $Z_{|0^m\rangle} = (2|0^m\rangle\langle 0^m| - I)$ to transform U_A into a Grover-like operator $W = Z_{|0^m\rangle}U_A$ which has the following matrix when restricted onto the invariant subspace S_λ in the $\{|0^m\rangle|\lambda\rangle, |\perp_{0^m,\lambda}\rangle\}$ basis

$$\begin{aligned} [W]_{\{|0^m\rangle|\lambda\rangle, |\perp_{0^m,\lambda}\rangle\}} &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} \lambda & \sqrt{1-\lambda^2} \\ \sqrt{1-\lambda^2} & -\lambda \end{pmatrix} \\ &= \begin{pmatrix} \lambda & \sqrt{1-\lambda^2} \\ -\sqrt{1-\lambda^2} & \lambda \end{pmatrix}, \end{aligned}$$

showing that W is still a $(1, m, 0)$-block-encoding of A. This has the form of a Y-axis rotation

$$[W]_{\{|0^m\rangle|\lambda\rangle, |\perp_{0^m,\lambda}\rangle\}} = \begin{pmatrix} \cos(\theta_\lambda) & \sin(\theta_\lambda) \\ -\sin(\theta_\lambda) & \cos(\theta_\lambda) \end{pmatrix},$$

[3] If $\lambda = \pm 1$, then there is no need for $|\perp_{0^m,\lambda}\rangle$, and the subspace S_λ becomes 1D.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

where $\theta_\lambda = \arccos(\lambda)$. Therefore, W has eigenvalues $e^{\pm i \arccos(\lambda)}$ with respective eigenvectors $(|0^m\rangle|\lambda\rangle \pm i|\perp_{0^m,\lambda}\rangle)/\sqrt{2}$, which can be accessed using quantum phase estimation.

Furthermore, we can see that on the span of the subspaces S_λ repeated application of W acts as

$$\begin{aligned} W^d &= \bigoplus_\lambda \begin{pmatrix} \cos(d\theta_\lambda) & \sin(d\theta_\lambda) \\ -\sin(d\theta_\lambda) & \cos(d\theta_\lambda) \end{pmatrix} \\ &= \bigoplus_\lambda \begin{pmatrix} T_d(\lambda) & \sqrt{1-\lambda^2}U_{d-1}(\lambda) \\ -\sqrt{1-\lambda^2}U_{d-1}(\lambda) & T_d(\lambda) \end{pmatrix} \\ &= \begin{pmatrix} T_d(A) & \cdot \\ \cdot & \cdot \end{pmatrix}, \end{aligned}$$

where $T_d(\cdot)$ and $U_d(\cdot)$ are degree-d Chebyshev polynomials of the first and second kind, respectively. Therefore, W^d applies the polynomial transformation T_d to each eigenvalue of A, thereby implementing $T_d(A)$.

Dominant resource cost (gates/qubits)

The resource cost of qubitization is inherited from the cost of the block-encoding. Given a Hermitian $(\alpha, m, 0)$-block-encoding U_A, the qubitization operator W is a (non-Hermitian) $(\alpha, m, 0)$-block-encoding, and it uses no additional qubits. The operation $Z_{|0^m\rangle} = (2|0^m\rangle\langle 0^m| - I)$ can be implemented (up to a global phase) with an m-qubit (anti)controlled $-Z$ gate, equivalent to an m-qubit Toffoli up to single-qubit gates. An example qubitization circuit is shown below in Fig. 10.5 for $m = 3$. Implementing a block-encoding of a degree-d Chebyshev polynomial applied to A requires d calls to U_A and $Z_{|0^m\rangle}$.

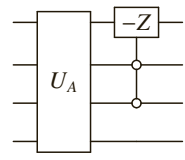


Figure 10.5 An example qubitization circuit using the Hermitian $(1, 3, 0)$-block-encoding U_A.

If the block-encoding U_A is not Hermitian, qubitization can be achieved using the construction of [717, Lemma 10] that uses one additional qubit, one call to controlled U_A, and one call to controlled $U_A^\dagger$ to implement the Hermitian

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

block-encoding

$$U'_A := ((HX) \otimes I)(|0\rangle\langle 0| \otimes U_A + |1\rangle\langle 1| \otimes U_A^\dagger)(H \otimes I)\,. \tag{10.5}$$

An alternative to qubitization is based on quantum singular value transformation (QSVT) that uses the sequence $Z_{|0^m\rangle} U_A^\dagger Z_{|0^m\rangle} U_A$, analogous to the earlier W^2, acting as

$$\begin{pmatrix} \lambda & \sqrt{1-\lambda^2} \\ -\sqrt{1-\lambda^2} & \lambda \end{pmatrix}^2$$

on a 2D subspace analogous to S_λ. The approach can be extended to odd-degree polynomials with a single additional application of $Z_{|0^m\rangle} U_A$ [431]. The advantage of this approach is that it does not require U_A to be Hermitian, thus there is no need for an additional qubit or calls to controlled $U_A^{\pm 1}$. This approach may be referred to as "quantum eigenvalue transformation" [687, 356] as this is a special case of QSVT applied to Hermitian A.

Caveats

The original formulation of qubitization [717] discussed above requires a Hermitian or normal block-encoded matrix A. The concept can be extended to general (non-square) matrices via QSVT, providing a significant generalization, however, in some cases quantum signal processing and its generalized versions [477, 255] can exploit additional structure that comes, for example, from the extra symmetries of Hermitian block-encodings, leading to potential constant-factor savings. Consider, for example, Hamiltonian simulation, where QSVT separately implements $\sin(tH)$ and $\cos(tH)$ using a block-encoding U_H of the Hamiltonian H, and applies a three-step oblivious amplification procedure on top of linear combination of unitaries to implement $\exp(itH)$ [431]. Meanwhile, quantum signal processing implements $\exp(itH)$ directly but requires an additional ancilla qubit and controlled access to a Hermitian block-encoding U'_H, which, when implemented via Eq. (10.5), uses both controlled U_H and $U_H^\dagger$, resulting in a factor of ~ 4 overhead. Altogether these considerations suggest that the QSVT-based approach might have a slightly better constant-factor overhead, particularly when controlled U_H is significantly more costly to implement than U_H. If U_H is already Hermitian, then quantum signal processing can have an improved complexity.

Example use cases

- Some quantum algorithms in quantum chemistry that compute energies perform phase estimation on a qubitization operator W implemented via calls to a block-encoding of the electronic structure Hamiltonian. This avoids the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

approximation error incurred when performing phase estimation on e^{iHt}, implemented via Hamiltonian simulation [841, 139].
- Qubitization acts as a precursor to QSVT, which extends the concept to general matrices and unifies it with quantum signal processing.

Further reading

- Original introduction of qubitization [717] and QSVT [431].
- A pedagogical overview of quantum signal processing, its lifting to QSVT, and their applications [744].
- Reference [687, Chapters 7 & 8] provides an accessible derivation of qubitization and QSVT.

10.5 Quantum singular value transformation

Rough overview (in words)

Quantum singular value transformation (QSVT) can be viewed as both a unification and generalization of qubitization and quantum signal processing. Given a block-encoding U_A of a general matrix A, QSVT enables the transformation of the singular values of A by a polynomial $f(\cdot)$. In QSVT, there is a one-to-one correspondence between the desired polynomial transformation and its quantum circuit implementation whose parameters can be found by efficient classical algorithms.

It transpires that a number of existing quantum algorithms have simple and optimal (or near-optimal) implementations via the QSVT framework, including Hamiltonian simulation [716, 717, 431], amplitude amplification and estimation [431, 855], quantum linear systems solving [431, 744], Gibbs sampling [431], algorithms for topological data analysis [514, 755, 143], and quantum phase estimation [744, 854].

Rough overview (in math)

We are given a $(1, m, 0)$-block-encoding U_A of operator A (for simplicity, we will restrict our presentation to square matrices A, noting there is a straightforward generalization to non-square A [431]), such that

$$A = (\langle 0^m| \otimes I) U_A (|0^m\rangle \otimes I),$$

where $|0^m\rangle$ denotes $|0\rangle^{\otimes m}$. The matrix A has a singular value decomposition (SVD)

$$A = \sum_i \sigma_i |w_i\rangle\langle v_i|.$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

QSVT provides a method for implementing

$$f^{(SV)}(A) := \begin{cases} \sum_i f(\sigma_i)|w_i\rangle\langle v_i| & \text{if } f \text{ is odd, and} \\ \sum_i f(\sigma_i)|v_i\rangle\langle v_i| & \text{if } f \text{ is even,} \end{cases}$$

for certain definite-parity polynomials $f\colon [-1,1] \to \mathbb{C}$, such that $|f(x)| \leq 1 \; \forall \; x \in [-1,1]$. Crucially, QSVT does not require us to know the SVD in advance; the transformation is carried out automatically by following an SVD-agnostic procedure outlined below. Note that $f^{(SV)}(A)$ only coincides with the matrix function $f(A)$ for Hermitian A (see §Caveats, below). In the Hermitian case, we can also obtain block-encodings of mixed-parity or complex functions by taking linear combinations of block-encodings—see [356] for examples.

By considering $U_A|0^m\rangle|v_i\rangle$ and $U_A^\dagger|0^m\rangle|w_i\rangle$ one can show that (see [687] for a step-by-step derivation) U_A and $U_A^\dagger$ act as linear maps between the 2D subspaces $S_i := \text{Span}\{|0^m\rangle|v_i\rangle, |\perp_i\rangle\}$ and $S_i' := \text{Span}\{|0^m\rangle|w_i\rangle, |\perp_i'\rangle\}$, with U_A being a transition matrix between these bases given by

$$\begin{array}{c} \\ |0^m\rangle|w_i\rangle \\ |\perp_i'\rangle \end{array} \begin{array}{c} \begin{array}{cc} |0^m\rangle|v_i\rangle & |\perp_i\rangle \end{array} \\ \begin{pmatrix} \sigma_i & \sqrt{1-\sigma_i^2} \\ \sqrt{1-\sigma_i^2} & -\sigma_i \end{pmatrix} \end{array}, \tag{10.6}$$

where both $|\perp_i\rangle, |\perp_i'\rangle$ are orthogonal to $|0^m\rangle$ (but not necessarily to each other).[4] The 2D subspace S_i is invariant under the operation $W := Z_{|0^m\rangle} U_A^\dagger Z_{|0^m\rangle} U_A$ (where $Z_{|0^m\rangle} = (2|0^m\rangle\langle 0^m| - I)$). The operation W, restricted onto the 2D subspace S_i, is written as

$$\begin{pmatrix} \sigma_i & \sqrt{1-\sigma_i^2} \\ -\sqrt{1-\sigma_i^2} & \sigma_i \end{pmatrix}^2.$$

An additional application of $Z_{|0^m\rangle} U_A$ maps back into the S_i' subspace. By analogy with qubitization, repeated application of W applies a Chebyshev polynomial to each of the singular values of A. In analogy with quantum signal processing, by lifting the $Z_{|0^m\rangle}$ reflection operation to a (controlled) rotation $e^{i\phi_j Z_{|0^m\rangle}}$ we can impose polynomial transformations of the singular values of A, which then induce the claimed polynomial transformation of A. It is typically convenient to use an additional ancilla qubit to implement $e^{i\phi_j Z_{|0^m\rangle}}$.

[4] If $\sigma_i = 1$, then there is no need for $|\perp_i\rangle, |\perp_i'\rangle$, and the subspaces S_i, S_i' become 1D.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

We define a QSVT circuit as the unitary sequence

$$U_\Phi := \begin{cases} e^{i\phi_1 Z_{|0^m\rangle}} U_A \prod_{j=1}^{(d-1)/2} \left(e^{i\phi_{2j} Z_{|0^m\rangle}} U_A^\dagger e^{i\phi_{2j+1} Z_{|0^m\rangle}} U_A \right) & \text{if } d \text{ is odd, and} \\ \prod_{j=1}^{d/2} \left(e^{i\phi_{2j-1} Z_{|0^m\rangle}} U_A^\dagger e^{i\phi_{2j} Z_{|0^m\rangle}} U_A \right) & \text{if } d \text{ is even,} \end{cases}$$

where $\Phi = (\phi_1, \phi_2, \ldots, \phi_d)$. We have that

$$(\langle 0^m| \otimes I) U_\Phi (|0^m\rangle \otimes I) = P^{(SV)}(A) = \begin{cases} \sum_i P(\sigma_i) |w_i\rangle\langle v_i|, & \text{for odd } d, \text{ and} \\ \sum_i P(\sigma_i) |v_i\rangle\langle v_i|, & \text{for even } d, \end{cases}$$

that is, the unitary U_Φ is a block-encoding of $P^{(SV)}(A)$, were P is the same polynomial that appears in quantum signal processing because the 2D matrix of Eq. (10.6) has the same form as the analogous 2D matrix in Eq. (10.4). We note that the constraints on the polynomials typically preclude direct implementation of the desired function as outlined above. By exploiting that $-\Phi$ implements P^*, we can use the circuit shown in Fig. 10.6 to implement a block-encoding of

$$P_{\Re}(A) = (\langle +| \otimes \langle 0^m| \otimes I)(|0\rangle\langle 0| \otimes U_\Phi + |1\rangle\langle 1| \otimes U_{-\Phi})(|+\rangle \otimes |0^m\rangle \otimes I)$$

for any definite-parity polynomial $P_{\Re} : [-1, 1] \to [-1, 1]$ by appropriately choosing Φ to implement a complex polynomial that fulfills the QSP conditions and then taking linear combinations of $U_\Phi, U_{-\Phi}$ to give a block-encoding of $P_{\Re}(A)$ [431, 744, 356].

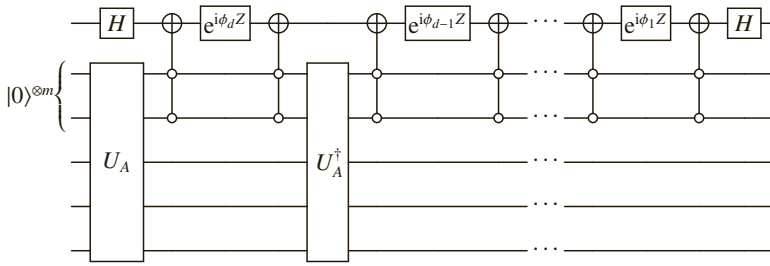


Figure 10.6 The QSVT circuit U_Φ which transforms a block-encoding U_A of A into a block-encoding of $f(A)$ for definite-parity $f : [-1, 1] \to [-1, 1]$ polynomial of degree d. As discussed in the main text, the angles $\{\phi_i\}$ can be calculated using efficient classical algorithms.

Dominant resource cost (gates/qubits)

Given a degree-d even-parity polynomial $f : [-1, 1] \to [-1, 1]$ and a $(1, m, 0)$-block-encoding U_A of A, one can implement a block-encoding of $f(A)$ using

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$d/2$ calls to U_A, $d/2$ calls to $U_A^\dagger$, $2d$ m-controlled Toffoli gates, and d single-qubit Z rotations (as shown in Fig. 10.6). Implementing a degree $d+1$ odd polynomial additionally requires another call to U_A, another two m-controlled Toffoli gates, and another single-qubit Z rotation. The QSVT circuit implements a $(1, m+1, 0)$-block-encoding of $f(A)$.

If U_A is imperfect (i.e., it is a $(1, m, \epsilon)$-block-encoding of A), then [431, Lemma 22] shows that the error in $f(A)$ is bounded by $4d\sqrt{\epsilon}$; that is, QSVT implements a $(1, m+1, 4d\sqrt{\epsilon})$-block-encoding of $f(A)$. Moreover, if the norm of A is bounded away from 1, for example, $\|A\| \leq 1/2$, then the perturbation bound can be improved to $O(d\epsilon)$ [431, Lemma 23].

Given an initial state $|\psi\rangle$, the success probability of implementing $f(A)|\psi\rangle$ is given by $|\langle\psi|f(A)^\dagger f(A)|\psi\rangle|^2$.

Caveats

Since the output must be subnormalized to ensure the existence of a unitary block-encoding of $f(A)$, f must satisfy $|f(x)| \leq 1 \; \forall \; x \in [-1, 1]$.

As noted above, $f^{(SV)}(A)$ is only guaranteed to coincide with the matrix function $f(A)$ for Hermitian A. As an example, choosing $f(x) = x^2$ we have $f^{(SV)}(A) = \sum_i \sigma_i^2 |v_i\rangle\langle v_i| = A^\dagger A$ whereas $A^2 = \sum_{i,j} \sigma_i \sigma_j |w_i\rangle\langle v_i|w_j\rangle\langle v_j|$. As discussed above, for the Hermitian case we can implement a block-encoding of a mixed-parity function f by taking linear combinations of block-encodings of its even and odd parts. However, in the general case when $|w_i\rangle$ and $|v_i\rangle$ do not coincide, it does not seem to be possible to remove the parity constraint, as the odd $\sum_i f_{\text{odd}}(\sigma_i)|w_i\rangle\langle v_i|$ and even $\sum_i f_{\text{even}}(\sigma_i)|v_i\rangle\langle v_i|$ singular value transforms potentially map to different subspaces.

As discussed in Section 10.3 on quantum signal processing, computing the angle sequence Φ can be a nontrivial classical task. Several approaches for accomplishing this task have been studied [477, 431, 255, 1065, 356, 1018, 359, 360, 19, 18, 782, 130], and researchers have reported computing angle sequences for polynomials up to $d \approx 10^7$ [782, 130].

As noted above, if $f(A)$ has small singular values, then preparing a quantum state $f(A)|\psi\rangle$ might require many repeated uses of its block-encoding, thus the normalization factor of f plays a crucial role in efficiency.

In many applications, one seeks to apply a function that is not a polynomial (e.g., e^x, e^{ix}, $\text{erf}(x)$). In such cases, one needs to first approximate the desired function by a polynomial (incurring an approximation error ϵ) in order to apply QSVT.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Example use cases

- Linear equation solving: Apply a polynomial approximation of $1/x$ to a block-encoding of $A^\dagger$ to get an approximate block-encoding of the pseudoinverse A^+.
- Hamiltonian simulation: Apply polynomial approximations of $\sin(x)$ and $\cos(x)$ to a block-encoding of a Hamiltonian H and combine them with linear combination of unitaries and amplitude amplification to obtain a block-encoding of e^{iHt}.
- Fixed-point amplitude amplification [1067]: Construct a polynomial that maps values in the domain $[a_{\min}, 1]$ to the range $[1-\delta, 1]$, and apply this polynomial to a state-preparation unitary that prepares the desired state with amplitude a. The result is amplification of the amplitude to at least $1-\delta$ as long as $a > a_{\min}$.
- For additional applications, see [431, 854, 744, 689, 688].

Further reading

- The QSVT framework was introduced in [431] and is also discussed in detail in [428].
- A pedagogical tutorial of the QSVT framework is given in [744, 687].
- A streamlined derivation of QSVT is presented in [980].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

11

Hamiltonian simulation

The task of Hamiltonian simulation is to approximately compile the evolution under a Hamiltonian $H(t)$, for time t, into a sequence of quantum gates. For a time-independent Hamiltonian, solving the Schrödinger equation (setting $\hbar = 1$) yields a time evolution operator $U(t) = e^{-iHt}$. In this chapter, we will discuss the equivalent operator $U(t) = e^{iHt}$, which is the more common definition in an algorithmic setting. We will assume $t \geq 0$, without loss of generality. The Hamiltonian of interest can arise from physical systems (e.g., quantum chemistry, condensed matter systems, or quantum field theories) but may also be constructed for other applications, such as differential equation simulation. Quantum simulation does not give full access to the amplitudes of the wavefunction during the simulation, unlike classical approaches based on exact diagonalization (or similar methods). Instead, we are only able to measure observables with respect to the time-evolved state, or use the state as an input to other quantum subroutines. Nevertheless, there are no known efficient classical methods that achieve this for general local or sparse Hamiltonians, suggesting an exponential quantum speedup. In fact, as a quantum computation can be expressed as a time evolution under a sequence of local (time-dependent) Hamiltonians, quantum simulation (i.e., time evolution and measurement of a given observable) is a BQP-complete problem.

Hamiltonian simulation algorithms require access to the Hamiltonian. There are three commonly used input models. The Pauli input model assumes that the Hamiltonian is given classically as a sum of products of Pauli operators, for example, $H = \sum_l h_l H_l$, where h_l are coefficients and H_l are multiqubit Pauli products. The d-sparse access model assumes that the Hamiltonian is a sparse matrix with at most d nonzero elements per row or column. We require that the locations of the nonzero elements and their values are efficient to compute classically. The density matrix access model assumes that the Hamiltonian corresponds to a density matrix, which we are either provided access to

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

copies of [708], or given a unitary that prepares a purification of the density matrix [717]. All of these input models can be used to prepare block-encodings of the Hamiltonian, which provides a standard-form access model that generalizes the above input models. Block-encodings are the input model of choice for some algorithms for Hamiltonian simulation (e.g., qubitization with quantum signal processing) [717].

Hamiltonian simulation can be used as a subroutine in a range of algorithms, including quantum phase estimation, quantum linear system solvers, Gibbs state preparation, and the quantum adiabatic algorithm. We remark that some of these algorithms are implicitly using Hamiltonian simulation to provide coherent, unitary access to the Hamiltonian. This can be particularly useful if few ancilla qubits are available, which may inhibit the use of some approaches to coherently access the Hamiltonian (e.g., block-encodings based on linear combinations of unitaries) but does not prevent the use of Hamiltonian simulation based on product formulas.

Each algorithm has its own advantages and disadvantages, as described at a high level in Table 11.1. Specific optimizations of each algorithm may be available for a given Hamiltonian. One can also consider hybridized methods combining two or more of the algorithms [718, 720, 820, 484, 852, 1027]. There are also other methods for Hamiltonian simulation, such as quantum walks [275, 133, 136] or density matrix–based Hamiltonian simulation [708, 619], which we do not discuss due to their less widespread use as algorithmic primitives for the applications discussed elsewhere in this book.

The authors are grateful to Yuan Su for reviewing this chapter.

11.1 Product formulas

Rough overview (in words)

Product formulas (or Trotter–Suzuki formulas/Trotterization) [705] are the most commonly used approach for Hamiltonian simulation, and are applicable to Hamiltonians in the Pauli access model and the sparse access model (see below for definitions of these models). Product formulas divide the evolution into a repeating sequence of short unitary evolutions under subterms of the Hamiltonian. These subterm evolutions have a known decomposition into elementary quantum gates. The error in product formulas depends on the commutators between different terms in the decomposition; if all of the terms in the Hamiltonian commute, product formulas are exact.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B881998BC0DA5A2CC99794A8C39

	Product formula (order k)	**qDRIFT**	**Taylor and Dyson series**	**QSP/QSVT**
# Qubits	$O(n)$	$O(n)$	$O(n + \log(L))$	$O(n + \log(L))$
Access model	Pauli Sparse	Pauli Sparse	Pauli Sparse Block-encoding	Block-encoding
Scaling	$O\left(5^{2k} nL \lVert H\rVert_1 t \left(\lVert H\rVert_1 t\epsilon^{-1}\right)^{\frac{1}{2k}}\right)$[a]	$O\left(n \lVert H\rVert_1^2 t^2 \epsilon^{-1}\right)$	$\widetilde{O}\left(\lVert H\rVert_1 tnL \log(\epsilon^{-1})\right)$	$O\left(nL(\lVert H\rVert_1 t + \log(\epsilon^{-1}))\right)$[b]
Pros	Commutator scaling. Simple implementation. Empirical performance. Minimal ancilla qubits.	L-independent scaling. No ancilla qubits.	$\log(1/\epsilon)$ scaling. Time-dependent simulations.	Optimal scaling with t, ϵ. Few ancilla qubits for algorithm.
Cons	Scaling with t, ϵ at low orders. Exponential prefactor (in order k).	Scaling with t, ϵ.	Many ancilla qubits if using noncompressed variant [718].	Time-dependent simulation. Ancilla/gate cost of block-encoding.

Table 11.1 High-level comparison of Hamiltonian simulation techniques. Quantitative comparisons assume a Pauli input model (which can easily be used to prepare a block-encoding of the Hamiltonian). For the stated complexity, we consider evolution $U(t) = e^{iHt}$ for time t under a time-independent Hamiltonian H on n qubits, given as a sum of L Pauli products $H = \sum_{j=1}^{L} h_j P_j$. The evolution is approximate to error ϵ in the spectral norm (diamond norm for qDRIFT). We define $\lVert H\rVert_1 = \sum_{j=1}^{L} |h_j|$. The qubit requirement for the Taylor and Dyson series method omits additional additive factors that scale logarithmically with the norm and/or derivative of the Hamiltonian. In specific applications it may be possible to reduce the number of qubits and/or gate complexity further by exploiting knowledge of the system, such as symmetries, commutation structure, or energy scales. For example, the factor of n present in the above complexities may be reduced by exploiting locality in the Pauli product terms of the Hamiltonian.

[a] The factor of n can be reduced to w when each Pauli term P_j acts nontrivially on at most w sites. The factor $\lVert H\rVert_1^{1+1/2k}$ can be reduced by exploiting commutativity of the various P_j.

[b] The factor nL derives from an upper bound on the gate complexity of block-encoding, and it can often be significantly improved by exploiting structure in h_j and H_j.

Product formula approaches have also been extended to treat time-dependent Hamiltonians [556, 1038, 1031, 34, 839]. In the following discussion, we will restrict our focus to the time-independent case, noting that the time-dependent approaches are executed in the same way, but have a slightly more complex error analysis.

Rough overview (in math)

Given a Hamiltonian H, desired evolution time t, and error ϵ, return a circuit $U(t)$ made of elementary gates such that

$$\|U(t) - \mathrm{e}^{\mathrm{i}Ht}\| \leq \epsilon.$$

In the above, we use the operator norm $\|\cdot\|$ (the maximal singular value) to quantify the quality of approximation, which controls the error for arbitrary input states (in trace distance) and for observables. This worst-case metric is mathematically convenient, but, as discussed below, tighter bounds may be obtained by using error metrics more closely aligned with the specification of the problem.

A product formula generates $U(t)$ through a product of easy-to-implement evolutions under terms in the Hamiltonian. For a Hamiltonian decomposition $H = \sum_{j=1}^{L} H_j$ with L terms, the first-order product formula with r steps is

$$S_1(t) = \left(\prod\nolimits_{j=1}^{L} \mathrm{e}^{\mathrm{i}H_j t/r}\right)^r.$$

The error in the first-order product formula is upper bounded as [286]

$$\|S_1(t) - \mathrm{e}^{\mathrm{i}Ht}\| \leq \frac{t^2}{2r} \sum_{i=1}^{L} \left\| \sum_{j=i+1}^{L} [H_i, H_j] \right\| \leq \frac{\|H\|_1^2 t^2}{2r},$$

where $\|H\|_1 = \sum_{j=1}^{L} \|H_j\|$. Higher-order formulas can be defined recursively, and are referred to as $(2k)$th-order product formulas. The error in a recursively defined $(2k)$th-order product formula is bounded by [286]

$$\|S_{2k}(t) - \mathrm{e}^{\mathrm{i}Ht}\| = O\left(\frac{\|H\|_1^{2k+1} t^{2k+1}}{r^{2k}}\right).$$

Product formulas can be applied to d-sparse Hamiltonians (at most d nonzero elements per row/column) with efficiently row computable nonzero elements [11]. Access to the nonzero elements of the Hamiltonian is provided via oracles O_f and O_H. The oracle O_f returns the column index (j) of the $k \in \{1, \ldots, d\}$th nonzero element in row i. The oracle O_H returns the value of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the matrix element H_{ij}.

$$O_f : O_f|k\rangle|i\rangle|0\rangle = |k\rangle|i\rangle|j\rangle$$
$$O_H : O_H|i\rangle|j\rangle|0\rangle = |i\rangle|j\rangle|H_{ij}\rangle.$$

Using graph-coloring algorithms, a d-sparse Hamiltonian H can be efficiently decomposed into a sum of efficiently simulable sparser Hamiltonians [134, 278]. Illustrating this idea with the approach of [134], we can decompose $H = \sum_{j=1}^{6d^2} H_j$, where each H_j is 1-sparse. The nonzero elements of a given H_j can be computed coherently using $O(\log^*(n))$ queries to O_f, O_H, where $\log^*$ is the iterated logarithm.[1] Time evolution under a 1-sparse Hamiltonian can be implemented efficiently using the approach of [11]. To simulate e^{iHt} using, for example, a first-order product formula, we sequentially apply each $e^{iH_j t}$ using the methods outlined above.

As a special case of the d-sparse access model, one can consider Hamiltonians given as a linear combination of L Pauli terms $H = \sum_{j=1}^{L} H_j = \sum_{j=1}^{L} \alpha_j P_j$, as each Pauli tensor product is already a 1-sparse matrix (so in this case, $d \leq L$). Time evolution under each Pauli term (or in some cases, groups of Pauli terms) can be simulated efficiently (see, e.g., [705, 801]), considerably simplifying the d-sparse construction by removing the need for oracles O_f and O_H.

Dominant resource cost (gates/qubits)

For an n-qubit Hamiltonian, product formulas act on n qubits. In the Pauli access model, no additional ancilla qubits are required. In the sparse access model, ancilla qubits may be required to implement the oracles O_f and O_H and to implement time evolution under 1-sparse Hamiltonians H_j.

The gate complexity is obtained by choosing the number of Trotter steps r sufficiently large to obtain an error ϵ and multiplying by the complexity of implementing each step of the product formula. It is necessary to balance the improved asymptotic scaling with t and ϵ of higher-order Trotter formulas against the exponentially growing prefactor of the higher-order formulas. In practical simulations of chemistry, condensed matter systems, or quantum field theories, a low-order formula (2nd–6th) typically minimizes the gate count.

A recursively defined $(2k)$th-order product formula (i.e., the first-order formula is given by $k = 1/2$, and is the base case) for simulating a d-sparse Hamiltonian for time t to accuracy ϵ requires [278]

$$O\left(5^{2k} d^2 (d + \log^* n)\|H\|t\left(\frac{d\|H\|t}{\epsilon}\right)^{1/2k}\right)$$

[1] For practical purposes, the iterated logarithm is essentially constant, since $\log^*(n) \leq 5$ for all $n \leq 2^{65536}$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

calls to the oracles O_f and O_H.

A recursively defined $(2k)$th-order product formula for simulating an L-term Hamiltonian in the Pauli access model for time t to accuracy ϵ requires [286]

$$O\left(5^{2k} nLt\left(\frac{t\alpha_{\text{comm},k}}{\epsilon}\right)^{1/2k}\right) \tag{11.1}$$

elementary single- and two-qubit gates, where

$$\alpha_{\text{comm},k} = \sum_{i_1,i_2,\ldots,i_{2k+1}} \|[H_{i_{2k+1}}, \ldots [H_{i_2}, H_{i_1}]]\| .$$

The dependence on $\alpha_{\text{comm},k}$ can be tightened and calculated for lower-order formulas (see [286] for full calculations). The dependence on n can be reduced to w for local Hamiltonians with Pauli terms that each act on at most w qubits.

Caveats

The error bounds of product formulas in the Pauli access model have been the object of significant investigation. Evaluating the tightest spectral norm bounds requires computing a large number of commutators between the terms in the Hamiltonian, which can be computationally intensive. Numerical simulations have shown that the commutator bounds can be loose by several orders of magnitude for chemical [72, 840] or spin [283] systems.

The spectral norm is the worst-case metric; it is an active area of research to find error metrics better suited to the problem at hand. For example, one may consider the *average*-case error over random input states [257, 1087] by the normalized Frobenius norm $\|U(t) - e^{iHt}\|_F / \sqrt{2^n}$. Recently, in [257] it was shown that the average-case error can be much smaller than the worst-case error for systems with large connectivity. More directly, one can also compute the Trotter error associated with input states from the low-energy [891, 516] or low-particle-number subspace [991, 963].

The gate counts of product formula approaches can also be reduced by grouping together mutually commuting terms such that they can be implemented using fewer gates than would be required to implement all the terms individually [124, 630, 225]. One can also reduce the number of Trotter steps required by randomizing the ordering of the terms [284, 288, 839] (although this must be balanced against any compilation benefits that may be obtained from a fixed ordering).

Example use cases

- Physical systems simulation: quantum chemistry, condensed matter systems, quantum field theories.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation, quantum adiabatic algorithm.

Further reading

- A rigorous derivation of the error in product formulas [286].
- A comparison of product formula methods with other approaches to Hamiltonian simulation for a concrete problem of interest [283].

11.2 qDRIFT

Rough overview (in words)

The *quantum stochastic drift protocol* [222], abbreviated as qDRIFT, operates in the Pauli access model[2] and approximates the Hamiltonian simulation channel (as opposed to the unitary) by randomly sampling a term from the Hamiltonian (according to the coefficient magnitudes), and then evolving under the chosen term. This process is repeated for a number of steps. Because it approximates the channel, rather than the unitary, it can be more difficult to use qDRIFT as a coherent subroutine in other algorithms (see §Caveats below).

The error in qDRIFT depends on the 1-norm of Hamiltonian coefficients. One main advantage of qDRIFT is that it does not explicitly depend on the number of terms in the Hamiltonian and has small constant overheads, making it well suited to systems with rapidly decaying interaction strengths, dominated by a few large terms. However, qDRIFT's time and error dependence are asymptotically worse than other methods, which seems to originate from the algorithm's randomized nature [258]. qDRIFT can also be extended to time-dependent Hamiltonian simulation with a Hamiltonian $H(t)$, where the gate count of the algorithm scales as $\int_0^t \|H(t')\| \mathrm{d}t'$, rather than as $t \max_{t'} \|H(t')\|$ like other Hamiltonian simulation algorithms [141]. We will restrict our discussion below to the time-independent case.

Rough overview (in math)

Given a Hamiltonian in the Pauli decomposition $H = \sum_i h_i H_i$ (with $\|H_i\| = 1$), qDRIFT provides a stochastic channel $\mathcal{N}$ that, when applied for N steps, approximates the Hamiltonian simulation channel

$$\|\mathcal{N}^N - \mathrm{e}^{\mathrm{i}Ht}(\cdot)\mathrm{e}^{-\mathrm{i}Ht}\|_\diamond \leq \epsilon$$

[2] qDRIFT was originally formulated, and is typically presented, for the Pauli access model [222], but the algorithm appears compatible with the d-sparse access model by applying it to the d-sparse decompositions in [134, 278] [961].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

to within diamond-norm error ϵ.

qDRIFT proceeds by randomly sampling terms according to their relative importance

$$X_k \overset{i.i.d.}{\sim} \frac{h_i H_i}{p_i}, \quad \text{where} \quad p_i = \frac{|h_i|}{\|H\|_1}$$

and $\|H\|_1 := \sum_i |h_i|$ is the sum of the strengths. Each step of qDRIFT then evolves the randomly sampled term X_k for a short period of time t/N, where N is a free parameter determining the number of qDRIFT steps, which controls the error in the simulation. The result is the following quantum channel:

$$\mathcal{N}[\rho] = \mathbb{E}[e^{i(t/N)X_k} \rho e^{-i(t/N)X_k}].$$

As discussed above, this channel is repeated for N steps, in order to approximate the Hamiltonian simulation channel.

Dominant resource cost (gates/qubits)

For an n-qubit Hamiltonian, qDRIFT acts on n register qubits, and no additional ancilla qubits are required.

In order to simulate the Hamiltonian evolution channel to within diamond-norm error ϵ, we require

$$N = O\left(\frac{\|H\|_1^2 t^2}{\epsilon}\right)$$

steps of qDRIFT [222, 258]. While the diamond-norm is a different error metric to the spectral norm used in other articles in this section, both metrics provide upper bounds on the error in an observable measured with respect to the time-evolved state [222]. For unitary channels, the diamond norm is effectively equal to the spectral norm (see, e.g., discussion in [480], up to constant factors).

The gate complexity is the number of steps multiplied by the individual costs of the elementary evolution $e^{i(t/N)X_k}$, which scales linearly with the locality of the Pauli operator X_k. When using qDRIFT to time evolve a state (e.g., for the purpose of measuring an observable), it is important to average the results over a sufficient number of independently sampled qDRIFT circuits [222].

Caveats

The qDRIFT algorithm has a quadratic dependence on time and a linear dependence on the inverse error $1/\epsilon$, while other Hamiltonian simulation methods can achieve linear time dependence and logarithmic inverse error dependence. A higher-order variant of qDRIFT was recently developed that improves the error dependence, but it is only suitable for estimating the expectation value of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

observables with respect to the time-evolved state, rather than approximating the unitary channel itself [794]. It is currently unclear how to design higher-order variants of qDRIFT that improve the time dependence, which appears to result from the randomized nature of the algorithm [258].

As discussed above, qDRIFT approximates the time evolution channel, rather than the unitary e^{iHt}. As a result, it can be difficult to incorporate as a subroutine in algorithms that seek to manipulate the unitary directly—for example, measuring $\mathrm{Tr}(U(t)\rho)$. Tasks of this form feature in some approaches for phase estimation [690], motivating alternate, qDRIFT-inspired approaches, in order to exploit qDRIFT-like benefits [1013].

Example use cases

- Physical systems simulation: quantum chemistry, condensed matter systems, quantum field theories.
- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation, quantum adiabatic algorithm.
- Hybridization with other quantum simulation methods [820, 852, 484].
- Using importance sampling to incorporate variable gate costs for simulating different terms X_k [621].

11.3 Taylor and Dyson series (linear combination of unitaries)

Rough overview (in words)

Taylor and Dyson series approaches for Hamiltonian simulation expand the time evolution operator as a Taylor series (time independent) [137] or Dyson series (time dependent) [615, 141] and use the linear combination of unitaries (LCU) primitive to apply the terms in the expansion, followed by (robust, oblivious) amplitude amplification to boost the success probability close to unity. These methods are close to being asymptotically optimal, achieving linear scaling in time and logarithmic dependence on the error. However, they use a large number of ancilla qubits, compared to other Hamiltonian simulation algorithms.

Rough overview (in math)

We focus on the time-independent case and follow the presentation in [137]. Given a Hamiltonian H, desired evolution time t, and error ϵ, return a circuit

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$U(t)$ made of elementary gates such that

$$\|U(t) - \mathrm{e}^{\mathrm{i}Ht}\| \leq \epsilon.$$

In the above, we use the operator norm (the maximal singular value) to quantify the worst-case error in the simulation.

The total evolution time t is divided into r segments. In each segment, we evolve under an approximation of $\mathrm{e}^{\mathrm{i}Ht/r}$. The Hamiltonian is decomposed into a linear combination of unitary operations $H = \sum_{l=1}^{L} \alpha_l H_l$, where we choose α_l real and positive by shifting phases into H_l, and $\|H_l\| = 1$. This decomposition appears naturally when the Hamiltonian is given as a linear combination of Pauli products. We approximate $\mathrm{e}^{\mathrm{i}Ht/r}$ using a Taylor expansion truncated to degree K

$$\begin{aligned}\mathrm{e}^{\mathrm{i}Ht/r} \approx U(t/r) &:= \sum_{k=0}^{K} \frac{1}{k!} (\mathrm{i}Ht/r)^k \\ &= \sum_{k=0}^{K} \sum_{l_1,\ldots,l_k=1}^{L} \frac{(\mathrm{i}t/r)^k}{k!} \alpha_{l_1} \ldots \alpha_{l_k} H_{l_1} \ldots H_{l_k}.\end{aligned}$$

Each segment $U(t/r)$ is implemented using robust oblivious amplitude amplification. Amplitude amplification is necessary because truncating the Taylor series at degree K makes $U(t/r)$ non-unitary. However, textbook amplitude amplification necessitates reflecting around the initial state (as well as the "good" state), which would be problematic since Hamiltonian simulation requires synthesizing a unitary that works simultaneously for all input states. This issue can be circumvented using oblivious amplitude amplification: we are given a unitary V such that for any state $|\psi\rangle$, we have $V|0^m\rangle|\psi\rangle = a|0^m\rangle U|\psi\rangle + b|0_\perp^m \phi\rangle$, for a unitary operator U, and the goal is to amplify the state $|0^m\rangle U|\psi\rangle$ to be obtained with probability 1 (we can recognize V as an $(a, m, 0)$ unitary block-encoding of U). A further problem is that the above operator $U(t/r)$ is non-unitary, and so deviates from the formulation of oblivious amplitude amplification [135]. The proven "robustness" property of oblivious amplitude amplification [137] ensures that the error induced by treating $U(t/r)$ as a probabilistically implemented unitary does not accumulate.

The value of K controls the error in the simulation and can be chosen as

$$K = O\left(\frac{\log(\|H\|_1 t/\epsilon)}{\log\log(\|H\|_1 t/\epsilon)}\right),$$

where we define $\|H\|_1 := \sum_{l=1}^{L} \alpha_l$. The total time evolution is divided into $r = \lceil \|H\|_1 t / \ln(2) \rceil$ segments, each of duration $\ln(2)/\|H\|_1$, which ensures that a

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

single application of robust oblivious amplitude amplification boosts the success probability of the segment to unity.

Within each segment, we apply $U(t/r)$ using the LCU primitive. This technique can be applied to Hamiltonians given in both the Pauli and d-sparse access models. For the Pauli access model, the Hamiltonian is already in the form of a linear combination of unitary operators. For the d-sparse case, we can use graph coloring algorithms [134, 278] to decompose the d-sparse Hamiltonian into a linear combination of unitaries, where each unitary is 1-sparse and self-inverse.

Dominant resource cost (gates/qubits)

In addition to the n-qubit data register, the Taylor series approach requires a number of ancilla registers to implement the LCU technique. In the original formulation [137], a register with K qubits is used to control the degree of the Taylor expansion, storing the value as $|k\rangle = |1^{\otimes k}0^{\otimes(K-k)}\rangle$. An additional K registers, each containing $\lceil \log_2(L) \rceil$ qubits, are used to index the possible values of each of the possible H_{l_k}. Hence, the overall space complexity of the original formulation [137] is $O\left(n + K\log(L)\right) = O\left(n + \log(\|H\|_1 t/\epsilon)\log(L)\right)$. In [718] it was shown how to reduce the space complexity to $O\left(n + \log(K) + \log(L)\right)$ using quantum counter circuits.

Additional ancilla qubits may be required to implement the LCU gadget (e.g., in the sparse access model) or for the reflections used in robust oblivious amplitude amplification.

As discussed above, implementing each segment requires one use of robust oblivious amplitude amplification, which makes two calls to the LCU circuit and one call to its inverse. The method incurs approximation errors from truncating the Taylor series at degree K and from the use of robust oblivious amplitude amplification. The resulting error per segment is bounded by $(e\ln(2)/(K+1))^{K+1}$.

The cost of the LCU circuit depends on the Hamiltonian access model. For the case of the Pauli access model, the LCU circuit requires two calls to a PREPARE operation that prepares the ancilla registers with the correct coefficients. In the compressed formulation [718], this requires $O(L+K)$ gates (compared to $O(LK)$ gates in the original formulation [137]). The LCU circuit also requires one call to a SELECT oracle, which can be implemented using K controlled select(H) operations. Each of these K operations can be implemented using $O(Ln)$ elementary gates [283, 75] (using quantum read-only memory).

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

The overall gate complexity in the Pauli access model is thus

$$O\left(\frac{\|H\|_1 tLn \log(\|H\|_1 t/\epsilon)}{\log\log(\|H\|_1 t/\epsilon)}\right) = \widetilde{O}\left(\|H\|_1 tLn \log\left(\frac{1}{\epsilon}\right)\right).$$

Using the LCU approach applied to a 1-sparse decomposition of a d-sparse Hamiltonian, the overall complexity is [137]

$$O\left(\frac{d^2\|H\|_{\max} tn \log^2(d^2|H|_{\max} t/\epsilon)}{\log\log(d^2\|H\|_{\max} t/\epsilon)}\right) = \widetilde{O}\left(d^2\|H\|_{\max} tn \log^2\left(\frac{1}{\epsilon}\right)\right),$$

where $\|H\|_{\max} = \max_{i,j}|\langle i|H|j\rangle|$. Using the amplification technique of [715], which utilizes quantum singular value transformation (QSVT), the dependence on d and $\|H\|_{\max}$ can be improved in some cases.

The extension to time-dependent Hamiltonians, through the use of a Dyson series, requires an additional "clock" register to store the time value and introduces a logarithmic dependence on the time derivative of the Hamiltonian [615, 141, 718].

Caveats

Concrete resource estimates for physical systems of interest have observed that the Taylor series approach may require more ancilla qubits and gates than product formulas or quantum signal processing approaches for Hamiltonian simulation [283], although these qubit counts would be improved by the subsequent compression approach to the algorithm [718]. The gate complexity of the algorithm can be reduced by exploiting anticommutativity in the Hamiltonian [1086], adding a corrective operation [804], or pruning terms with small magnitudes from the expansion [759].

Example use cases

- Physical systems simulation: quantum chemistry (see [73, 74, 962, 718]), condensed matter systems, quantum field theories.
- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation, quantum adiabatic algorithm.
- Hamiltonian simulation in the interaction picture [718].

Further reading

- A comparison of several Hamiltonian simulation algorithms, including Taylor series [283].
- Derivations of the compressed variants of Hamiltonian simulation via Taylor/Dyson series [718, Appendices B & D].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

11.4 Quantum signal processing / quantum singular value transformation

Rough overview (in words)

Quantum signal processing (QSP) and quantum singular value transformation (QSVT) are techniques for applying polynomial transformations to block-encoded operators. These techniques can be used to implement Hamiltonian simulation, given a block-encoding of the Hamiltonian. Both approaches have optimal scaling with t and ϵ for time-independent Hamiltonians.

QSP was initially developed for the d-sparse access model [716]. Through the introduction of block-encodings and qubitization, it was made applicable in a standard form to Hamiltonians in a Pauli access model, d-sparse access model, or given as density matrices (where we are given access to a unitary that prepares a purification of the density matrix) [717]. QSVT was later developed as a more general and direct route to the results of QSP [429].

Hamiltonian simulation via QSP or QSVT is less well suited to time-dependent Hamiltonians, as the need to Trotterize the time-dependent evolution breaks the optimal dependence on the parameters.

Rough overview (in math)

Access to the Hamiltonian H is provided by an $(\alpha, m, 0)$-block-encoding U_H (the case of approximate block-encodings can be treated using [429, Lemma 22]) such that

$$(\langle 0^m| \otimes I) U_H (|0^m\rangle \otimes I) = H/\alpha.$$

The Hamiltonian has a spectral decomposition of $\sum_\lambda \lambda |\lambda\rangle\langle\lambda|$. We seek to use U_H to implement an operator $U(t)$ approximating

$$\|U(t) - \sum_\lambda \mathrm{e}^{\mathrm{i}\lambda t} |\lambda\rangle\langle\lambda|\| \leq \epsilon.$$

Qubitization converts U_H into a more structured unitary W (which is also a block-encoding of the Hamiltonian). The eigenvalues of W are $\mathrm{e}^{\pm \mathrm{i} \arccos(\lambda/\alpha)}$, directly related to those of H. QSP then enables polynomial transformations to be applied to these eigenvalues, which defines the application of the polynomial to W. This concept can be generalized via QSVT, which effectively unifies the qubitization and QSP step.

In both cases, our goal is to implement a block-encoding of $U(t) \approx \sum_\lambda \mathrm{e}^{\mathrm{i}\lambda t} |\lambda\rangle\langle\lambda|$, which defines Hamiltonian simulation. In QSVT we separately implement polynomials approximating $\cos(\lambda t)$ and $i \sin(\lambda t)$, combine them using a linear combination of block-encodings, and boost the success probability using three-step oblivious amplitude amplification. Further details

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

can be found in [429, 744]. Meanwhile, QSP implements $\exp(itH)$ directly but requires an additional ancilla qubit and controlled access to a Hermitian block-encoding U'_H, which, when implemented via Eq. (10.5), uses both controlled U_H and $U_H^\dagger$ resulting in a factor of ~ 4 overhead [717]. Altogether, these considerations suggest that the QSVT-based approach might have a slightly better complexity, particularly when controlled U_H is significantly more costly to implement than U_H. If U_H is already Hermitian, then QSP can have a lower complexity.

Dominant resource cost (gates/qubits)

Using either QSP or QSVT, block-encoding a degree-k polynomial $f(H)$ is performed using $O(k)$ calls to the block-encoding U_H [717, 429]. Hence, the degree of the polynomial approximating e^{iHt} determines the complexity of Hamiltonian simulation using these techniques. As noted in [429, Corollary 60], we can rigorously bound the resources for Hamiltonian simulation via QSVT for all values of t as using

$$O\left(\alpha t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/\alpha t)}\right)$$

calls to the $(\alpha, m, 0)$-block-encoding U_H. This query complexity is optimal [134, 429], although the block-encoding can hide additional complexities, in practice. In some cases, the dependence on norm parameters can be improved by exploiting details of the simulated system; see [715, 714].

For a Pauli access model, the block-encoding is implemented using the linear combination of unitaries (LCU) primitives PREPARE and SELECT. For a Hamiltonian with L terms $\alpha = \|H\|_1$, $m = O(\log(L))$, and two additional qubits are required for QSVT. The overall gate complexity depends on the exact implementation of PREPARE and SELECT, which can often be tailored to the Hamiltonian of interest. In the worst case, PREPARE uses $\Theta(L)$ gates, and SELECT uses $\Theta(nL)$ gates (although these can be significantly improved by exploiting structure in the Hamiltonian; see, e.g., [75, 1011]). Thus, the overall worst-case gate complexity is

$$O\left(nL\left(\|H\|_1 t + \frac{\log(1/\epsilon)}{\log(e + \log(1/\epsilon)/\|H\|_1 t)}\right)\right).$$

For a d-sparse access model, $\alpha = d\|H\|_{\max}$, where $\|H\|_{\max} = \max_{i,j}|\langle i|H|j\rangle|$, $m = O(\log(d))$, and two additional qubits are required for QSVT. The overall gate complexity depends on the cost of sparse access to elements of H. Assuming a circuit for sparse access with constant gate complexity, the overall

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

gate complexity is

$$O\left(d\|H\|_{\max}t + \frac{\log(1/\epsilon)}{\log(\mathrm{e} + \log(1/\epsilon)/d\|H\|_{\max}t)}\right).$$

Using the QSVT-based amplification technique of [715], the dependence on $d\|H\|_{\max}$ can be improved in some cases.

The density matrix access model seeks to perform time evolution under $\mathrm{e}^{\mathrm{i}\rho t}$, given access to either multiple copies of ρ or a unitary U_ρ that prepares a purification of ρ. Given U_ρ, we can prepare a block-encoding of ρ [717] (see Section 10.1 on block-encodings for details) with $\alpha = 1$. If the gate complexity of U_ρ is $C(U_\rho)$, then the overall gate complexity is

$$O\left(C(U_\rho)\left(t + \frac{\log(1/\epsilon)}{\log(\mathrm{e} + \log(1/\epsilon)/t)}\right)\right).$$

Caveats

The method was found to perform competitively with Trotterization (and better than Taylor series) in concrete resource estimates for simulating spin-chain Hamiltonians [283]. While that work had difficulty calculating the QSP phase factors, this issue has since been addressed with the development of classical algorithms for finding the phase factors (e.g., [431, 477, 356, 255], and successors). Nevertheless, this contributes a classical preprocessing cost to the algorithm.

It is currently unclear how to perform optimal time-dependent Hamiltonian simulation with these methods, without resorting to Trotterization. Some initial investigations have shown promising results using clock Hamiltonian constructions [1027] or for time-periodic Hamiltonians [770, 769].

Example use cases

- Physical systems simulation: quantum chemistry, condensed matter systems (see [283]), quantum field theories, differential equations in plasma physics (see [803]).
- Algorithms: quantum phase estimation, quantum linear system solvers, Gibbs state preparation.

Further reading

- Pedagogical overviews [744, 687].
- Comparison of several Hamiltonian simulation algorithms [283].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

12

Quantum Fourier transform

The authors are grateful to Ronald de Wolf for reviewing this chapter.

Rough overview (in words)

The quantum Fourier transform (QFT) is a quantum version of the discrete Fourier transform (DFT) and takes quantum states to their Fourier-transformed version.

Rough overview (in math)

The QFT is a quantum circuit that takes pure N-dimensional quantum states $|x\rangle = \sum_{i=0}^{N-1} x_i|i\rangle$ to pure quantum states $|y\rangle = \sum_{i=0}^{N-1} y_i|i\rangle$ with the Fourier-transformed amplitudes

$$y_k = \frac{1}{\sqrt{N}} \sum_{l=0}^{N-1} x_l \exp(2\pi \mathrm{i} kl/N) \quad \text{for } k = 0, \ldots, N-1. \tag{12.1}$$

Dominant resource cost (gates/qubits)

The space cost is $O(\log(N))$ qubits and the quantum complexity of the textbook algorithm is $O(\log^2(N))$. In terms of Hadamard gates, swap gates, and controlled phase shift gates $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes R_\ell$ with

$$R_\ell = \begin{pmatrix} 1 & 0 \\ 0 & \exp\left(2\pi \mathrm{i} 2^{-\ell}\right) \end{pmatrix},$$

the quantum circuit is given in Fig. 12.1 (see also [801, Fig. 5.1]), where $N = 2^n$. The swap gates at the end of the circuit are required to reverse the order of the output qubits. The complexity can be improved to

$$O\left(\log(N) \log\left(\log(N)\epsilon^{-1}\right) + \log^2\left(\epsilon^{-1}\right)\right)$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

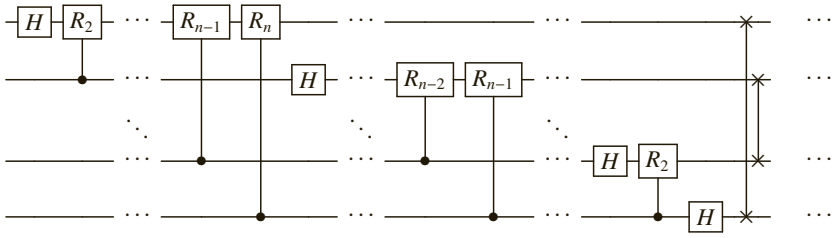


Figure 12.1 Quantum circuit implementation of QFT.

when only asking for ϵ-approximate solutions [486]. Finite constants and compilation cost for fault-tolerant quantum architectures are also discussed in the literature. For example, [795] gives an implementation with $O(\log(N)\log\log(N))$ T gates and estimates finite T-gate costs for different instance sizes.

Caveats

- The QFT does not achieve the same task as the classical DFT that takes vectors $(x_0, \ldots, x_{N-1}) \in \mathbb{C}^N$ to vectors $(y_0, \ldots, y_{N-1}) \in \mathbb{C}^N$ with y_k defined as in Eq. (12.1). The DFT can be implemented via the fast Fourier transform in classical complexity $O(N\log(N))$, which is exponentially more costly than the quantum complexity $O(\log^2(N))$ of the QFT. However, for the QFT to achieve the same task as the DFT, pure state quantum tomography would be required to read out and learn the Fourier-transformed amplitudes, which destroys any quantum speedup for the DFT.
- When QFT is employed in use cases, for example, for factoring, one has to be careful in finite-size instances when counting resources [943], and for this a semiclassical version of the QFT can be more quantum resource efficient [458].
- The QFT admits an efficient representation as a matrix product operator (a type of tensor network), meaning that the approximation improves exponentially in the bond dimension [262]. This suggests that quantum algorithms relying on the QFT for speedup must involve highly entangled input or intermediate states, in order to beat state-of-the-art tensor network methods.

Example use cases

- Even though the QFT does not speedup the DFT, QFT is used as a subroutine in more involved quantum routines with large quantum speedup. Examples include quantum algorithms for the discrete logarithm problem, the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

hidden subgroup problem, the factoring problem, to name a few. The QFT can be seen as the crucial quantum ingredient that allows for a superpolynomial end-to-end quantum speedup for these problems. We discuss this in the context of quantum cryptanalysis in Chapter 6.

- The QFT appears in the standard circuit for quantum phase estimation, where it is used to convert accrued phase estimation into a binary value that can be read out.
- The QFT is used for switching between the position and momentum bases in grid-based simulations of quantum chemistry [601] or quantum field theories [588].

Further reading

- Textbook reference [801, Section 5.1].
- The quantum Fourier transform can be generalized to other groups. The version presented above is for the group $\mathbb{Z}/(2^n\mathbb{Z})$. Its implementation for other abelian groups as well as nonabelian groups is discussed in [277] and the references therein.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

13

Quantum phase estimation

The authors are grateful to Patrick Rall and Ronald de Wolf for reviewing this chapter.

Rough overview (in words)

The quantum phase estimation (QPE) subroutine produces an estimate of an eigenvalue of a unitary operator. It is a cornerstone of quantum algorithms primitives and has numerous applications. For example, Shor's algorithm for factoring can be viewed as an application of QPE together with modular exponentiation. Similarly, when combined with Hamiltonian simulation, QPE can produce an estimate for an eigenvalue of a Hamiltonian (given an appropriate initial state), an important problem in areas such as quantum chemistry. Generally, since quantum computations enact unitary operators, quantum phase estimation is an essential algorithmic tool for accessing information about these operators, specifically, information about their periodicities, and the properties of their eigenstates.

As one of the oldest quantum primitives discovered [624, 299], QPE has played a significant historical role in the development of quantum algorithms. In a typical use case, QPE is used as a first step to compute an estimate of the eigenvalue of the unitary into an ancilla register. Then, the ancilla register is used as a control for subsequent operations. However, in some applications, such as Gibbs sampling and solving the quantum linear system problem, this procedure must be applied *coherently* to a superposition of eigenstates with different eigenvalues, and the estimate of the eigenvalue must be uncomputed at the end. As discussed below, coherent usage of the QPE primitive in this manner must be handled with care, due to several identified caveats. While QPE still provides essential intuition for how these applications work, in some

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

cases, modern techniques leveraging quantum signal processing and the quantum singular value transformation [431] lead to a cleaner and more direct analysis than QPE.

In the discussion below, we begin with the textbook presentation of QPE [299, 801], and expound on the aforementioned caveats. We also present example use cases, noting the instances where QPE was originally a key ingredient but no longer features directly in state-of-the-art solutions.

Rough overview (in math)

Let U be a unitary with eigendecomposition $U = \sum_j \mathrm{e}^{\mathrm{i}2\pi\phi_j}|\psi_j\rangle\langle\psi_j|$. Given as input the state $|\psi_j\rangle$, the QPE subroutine produces an estimate $\hat{\phi}_j$ for ϕ_j. The algorithm requires the ability to apply controlled U^{2^p} for non-negative integers p. If ϕ_j is an exact multiple of 2^{-P}, then an exact estimate of ϕ_j can be learned with certainty using only $p \in \{0, 1, \ldots, P-1\}$. In general, an estimate $\hat{\phi}_j$ of ϕ_j satisfying $|\phi_j - \hat{\phi}_j| \leq \epsilon$ can be learned with high probability by taking the maximum value of 2^p on the order of $1/\epsilon$. The algorithm also requires application of an inverse quantum Fourier transform to orchestrate the constructive interference near the estimate for ϕ_j. The quantum circuit for the standard approach to QPE is shown in Fig. 13.1.

Phase estimation can also be applied coherently onto a superposition of eigenstates. Suppose that the input state is $|\psi\rangle = \sum_j \alpha_j|\psi_j\rangle$. By linearity, if each phase ϕ_j is a multiple of 2^{-P} and phase estimation is run with sufficient resolution, then QPE enacts the following unitary

$$|\psi\rangle|0\rangle \mapsto \sum_j \alpha_j|\psi_j\rangle|\phi_j\rangle, \tag{13.1}$$

where $|\phi_j\rangle$ holds a P-bit binary representation of ϕ_j. If the auxiliary register is measured—here assuming for simplicity that the eigenvalues ϕ_j are nondegenerate—then with probability $|\alpha_j|^2$ (consistent with the Born rule) the estimate ϕ_j is obtained and the state collapses to the corresponding eigenstate $|\psi_j\rangle$.[1] If the phases ϕ_j are not multiples of 2^{-P}, an approximate version of this operation can still be accomplished as long as the precision is sufficiently small to resolve the eigenvalues, subject to some caveats (discussed below).

[1] Alternatively, if ϕ_j is known ahead of time (to sufficient precision), QPE can be wrapped inside of amplitude amplification and the state $|\psi_j\rangle$ can be prepared using $O(|\alpha_j|^{-1})$ applications of the QPE circuit, rather than $O(|\alpha_j|^{-2})$. Note that amplitude amplification can be understood through the QSVT [431] formalism, and in many applications, such as projecting onto the ground state of a Hamiltonian [688], one can achieve this sort of scaling directly without explicitly relying on QPE.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

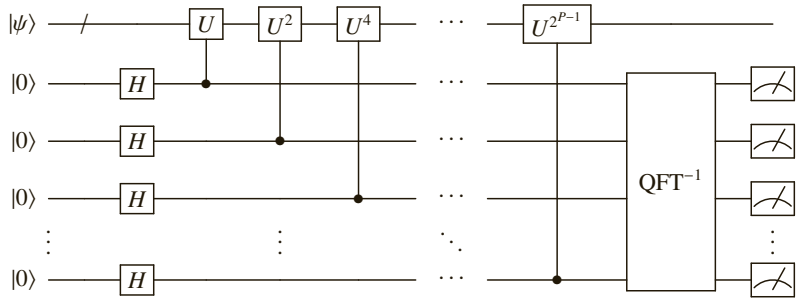


Figure 13.1 Quantum circuit implementation of QPE. The measurement outcomes on the P ancilla qubits give a P-bit estimate of the phase ϕ_j (correct up to error $O(2^{-P})$) with high probability.

Dominant resource cost (gates/qubits)

The QPE subroutine is typically dominated by calls to the controlled unitary U. If resolution ϵ is desired, one must perform controlled U^{2^p} operations for $p \in \{0, 1, \ldots, \lceil \log_2(1/\epsilon) \rceil + O(1)\}$; thus, the number of calls to a controlled U oracle will be $O(1/\epsilon)$. This dependence on ϵ is optimal; the $O(1/\epsilon)$ scaling is known as the *Heisenberg limit*.

In the context of estimating the eigenenergy of a Hamiltonian H, one can choose $U = e^{iH}$, and then implement controlled U^t, that is, controlled e^{iHt}, with Hamiltonian simulation. In this case, given the ability to prepare an eigenstate of H, an ϵ-approximation of the eigenvalue requires values of t up to $O(1/\epsilon)$.[2] However, one must also factor in the error in the Hamiltonian simulation. In a typical setting, access to the n-qubit Hamiltonian is given through a linear combination of L unitaries, for instance, Pauli matrices. Let $\|H\|_1$ denote the sum of the coefficients in the combination. Then, methods for Hamiltonian simulation based on quantum signal processing can approximate e^{iHt} to error $O(\epsilon)$ with $O(nL(\|H\|_1 t + \log(1/\epsilon)))$ gate complexity, whereas methods based on product formulas incur cost $O(nL(\|H\|_1 t)^{1+1/2k}\epsilon^{-1/2k})$ for $(2k)$th-order product formulas, although the actual cost can be lower after accounting for structure in the Hamiltonian terms. Balancing the error from phase estimation against the error from Hamiltonian simulation can cause sub-Heisenberg-limited performance, such as in the case of the product formula approach. The overhead associated with imperfect Hamiltonian simulation can be avoided by applying QPE to different functions of H; for example, a promising choice is the qubitization

[2] The fact that learning energies to greater precision requires a proportionally greater amount of time t is a manifestation of the energy-time Heisenberg uncertainty principle, and forms the origin of the term "Heisenberg limit."

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

operator, which acts in a similar way to $U = e^{i \arccos(H/\alpha)}$, where α is the normalization factor of the qubitization operator. The reason this is advantageous is that the qubitization operator can be implemented *exactly* given access to a block-encoding of H [717, 841, 139]. In general, we require the unitary U on which phase estimation is performed to be a known, classically invertible function of the Hamiltonian $U = f(H)$. The complexity of QPE depends on the desired uncertainty in the energy eigenvalue, which can be related to the uncertainty in the measured eigenphase via the magnitude of the derivative of the function, $\|f'(\cdot)\|$.

The number of qubits for QPE is simply the size of the register needed to hold the input state $|\psi_j\rangle$ plus the size of the register needed to hold the estimate $\hat{\phi}_j$ (i.e., roughly $\lceil \log_2(1/\epsilon) \rceil$ bits). Additionally, QPE requires an inverse quantum Fourier transform (QFT), which (using the textbook QFT implementation) adds only $O(\log^2(1/\epsilon))$ additional gates to the protocol.

Another version of QPE [627] achieves the same task with only a single ancilla qubit, but, as a result, learns only one bit of the output at a time. Additionally, it requires an exact eigenstate as input. The latter problem can be avoided using a statistical approach [690, 1013].

Caveats

The main caveats of QPE are related to the fact that eigenphases are not always exact integer multiples of 2^{-P}, resulting in noncertain outcomes of QPE, which can lead to complications in certain applications.

- Fat tails and boosting of success probability: Whenever the phases ϕ_j are not exact integer multiples of 2^{-P} for some integer P, phase estimation will not return the answer ϕ_j with certainty. Rather, there will be a distribution of possible estimates $\hat{\phi}_j$ that is peaked near ϕ_j. If one chooses $P = \lceil \log_2(1/\epsilon) \rceil + O(1)$, then most of the probability mass of this distribution lies within ϵ of ϕ_j. As P is increased further, the distribution becomes more sharply peaked near ϕ_j, and if an ϵ-accurate estimate with $1 - \delta$ probability is desired, one must take $P = \lceil \log_2(1/\epsilon) \rceil + O(\log(1/\delta))$, corresponding to a multiplicative $O(1/\delta)$ overhead in the query complexity to U and $O(\log(1/\delta))$ additional ancilla qubits. This poor δ dependence is due to "fat tails" on the distribution of estimates of $\hat{\phi}_j$. One way to avoid this overhead is to take the median of estimates obtained from $O(\log(1/\delta))$ repetitions of QPE [792, Lemma 1]. A downside of this approach is that it may be difficult to implement coherently on a superposition of eigenstates, in the sense of Eq. (13.1), since computing the median would require a coherent quantum sorting network. An alternative way to circumvent the fat tails problem

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

is to modify the QPE protocol to have a nonuniform superposition in the register that controls applications of U; a judicious choice of superposition leads the distribution over estimates $\hat{\phi}_j$ to be a Kaiser window (see [143, Appendix D] and [144]) or discrete prolate spheroidal sequence (DPSS) function [829], which minimizes the probability of deviating from ϕ_j by more than ϵ. See also [260], where a Gaussian profile is used to suppress the tails. Boosting the success probability to $1 - \delta$ in this fashion incurs multiplicative $O(\log(1/\delta))$ cost, rather than $O(1/\delta)$. The overall cost in queries to U by these methods matches a lower bound of $\Omega(\epsilon^{-1} \log(1/\delta))$ shown in [736].

- Performing coherent QPE: When ϕ_j are noninteger multiples of 2^{-P}, the coherent operation in Eq. (13.1) cannot be straightforwardly performed with exact fidelity. This is because for each value of j, the second register will be in a superposition of many values of $\hat{\phi}_j$ (most but not all of the amplitude will lie on estimates close to ϕ_j). To restore coherence, one might try coherently rounding the estimate $\hat{\phi}_j$ onto a coarser net of grid points (and then uncomputing the original estimate $\hat{\phi}_j$); however, there will always be edge cases where ϕ_j falls very near the midpoint between two grid points and rounding destroys some of the coherence in the input. This is true even as the precision of QPE is taken to zero ($\epsilon \to 0$). See [854] for a discussion. One possible way to mitigate this issue is presented in the "consistent phase estimation" protocol of [975, Section 5.2], where a random shift is applied to the grid points to avoid this situation for any particular eigenphase with high probability. However, this does not generically work simultaneously for all eigenphases. In [854], it is shown that performing the map of Eq. (13.1) is impossible without a "rounding promise" on the set of eigenphases $\{\phi_j\}$.
- Biased estimator: A further consequence of the noncertainty of the QPE output is that the estimate $\hat{\phi}_j$ is *biased*; that is, its expectation value is not exactly equal to ϕ_j. This issue can also be fixed with a random shift idea, yielding an unbiased (and symmetrically distributed) version of QPE [691, 49].

Example use cases

- In quantum chemistry and condensed matter physics, QPE can be used to measure the eigenvalues (and especially the ground state energy) of the Hamiltonian H, which gives knowledge about reaction mechanisms, stable configurations, and other equilibrium properties. For QPE to succeed, a trial state $|\psi\rangle$ with substantial overlap with the eigenstate of interest must be input to QPE, which is challenging in the general case. The problem of ground state preparation has garnered intense study, and state-of-the-art techniques do not always follow the textbook method that relies on the QFT, presented

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

above. For example, quantum signal processing can be leveraged directly to filter out unwanted eigenstates [689, 688], effecting a similar outcome as QPE.

- In Shor's algorithm, given a composite integer N and a (randomly chosen) base $x < N$, QPE is used to determine the order of x, that is, the minimum integer r for which $x^r \equiv 1 \mod N$, which is in turn used to infer the prime factors of N. Here, the unitary U is the modular multiplication unitary that sends $|y\rangle \mapsto |xy \mod N\rangle$.
- In amplitude estimation [186], given a unitary U that prepares a state $U|\psi_0\rangle = a|\psi_g\rangle + b|\psi_b\rangle$, QPE is used to estimate $|a|$ or $|a|^2$. More advanced approaches to amplitude estimation not relying on QPE have since been developed. These leverage Grover's algorithm, or more generally quantum signal processing, without using the QFT. While these do not surpass the QPE-based method in asymptotic complexity, they potentially offer other benefits, such as improved practical performance and versatility. See [855] and references therein.
- In the Monte Carlo–style quantum algorithms for Gibbs sampling of quantum (i.e., nondiagonal in the computational basis) Hamiltonians, roughly speaking, the quantum state undergoes a random walk on the eigenbasis of the Hamiltonian. Steps of this random walk are accepted or rejected according to how much the energy changes at each step. The QPE subroutine is used to simultaneously (approximately) project onto the eigenbasis of the Hamiltonian and to produce an estimate of the energy, used to determine whether the step should be accepted or rejected. Early studies [984, 1076, 1048] of this approach were hampered by the caveats related to rejecting quantum states and imperfect energy estimates, but recent works [856, 260, 259] circumvent these problems (by randomizing the grid points or completely abandoning phase estimation).
- To follow the ground state $|\psi_0(s)\rangle$ of a Hamiltonian $H(s)$ as some parameter s is varied from 0 to 1, one can run the adiabatic algorithm. Alternatively, one can consider a discretization of steps $s_t \in \{s_0, \ldots, s_T\}$, where $0 = s_0 < s_1 < s_2 < \cdots < s_{T-1} < s_T = 1$, and run QPE on $H(s_t)$ in succession, each time causing a measurement in the eigenbasis of $H(s_t)$. Due to the quantum Zeno effect, as long as sufficiently small steps are taken, each projection will be onto the ground space with high probability (see, e.g., [944]). Larger steps can be tolerated if one boosts the probability that each step succeeds with amplitude amplification [162]. This approach is similar to the idea in Hastings' short-path algorithm [505, 329], which solves combinatorial optimization problems. However, note that modern implementations along these lines would likely elect to perform the ground state projection

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

via eigenstate filtering [689] or related QSVT-based methods, rather than QPE.

- While state-of-the-art quantum linear system solvers (QLSSs) do not explicitly use QPE, the original QLSS by Harrow, Hassidim, and Lloyd [500] uses QPE to coherently measure the eigenvalues of a matrix A into an auxiliary register. These eigenvalue estimates are subsequently inverted with coherent arithmetic in order to produce the state $A^{-1}|b\rangle$ corresponding to the solution to the system $Ax = b$. Achieving optimal asymptotic performance requires additional ingredients beyond QPE, and is best understood through the language of block-encodings and quantum linear algebra. This framework allows for manipulation of eigenvalues without explicitly reading them into an ancilla register with QPE.
- In certain machine learning tasks related to linear algebra, such as principal component analysis [708] and recommendation systems [608], quantum algorithms have been proposed that leverage QPE to access the information about the eigenvectors and eigenvalues. As explained in [854], these have not always fully accounted for the caveat related to coherent QPE, although typically these caveats can be circumvented using the framework of quantum linear algebra [248, 431].

Further reading

- The standard circuit and analysis of QPE appears in Nielsen and Chuang [801]. See also [299].
- Many variants of the QPE algorithm have been explored, which can be superior to the standard version in certain settings. See, for example, [854, 690] for additional references and informative overviews of various methods, along with their advantages and drawbacks.
- Reference [687] contains a pedagogical overview of QPE including some of its variants and applications.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

14

Amplitude amplification and estimation

Quantum amplitude amplification and estimation provide means to boost or extract the amplitude of a marked quantum state that is produced in superposition with orthogonal states by a unitary matrix. They are among the most widely used quantum primitives, providing quadratic speedups over classical algorithms in many settings.

The authors are grateful to Patrick Rall for reviewing this chapter.

14.1 Amplitude amplification

Rough overview (in words)

Given a quantum subroutine that succeeds with a probability less than one, amplitude amplification can be used to boost the success probability to 1 by making repeated calls to the subroutine and to a unitary that determines if the subroutine has succeeded. Amplitude amplification can be viewed as a generalization of Grover's search algorithm [464] and offers a quadratic speedup compared to classical methods in many instances.

Rough overview (in math)

We are given an initial state $|\psi_0\rangle$, a target ("good") state $|\psi_g\rangle$ that we can mark (i.e., the ability to reflect about the state), and a unitary U (and its inverse $U^\dagger$) such that

$$U|\psi_0\rangle = |\psi\rangle = a|\psi_g\rangle + b|\psi_b\rangle\,,$$

where $|\psi_b\rangle$ is a ("bad") state orthogonal to the target state. In other words, $|a|^2$ is the probability of success of applying U and measuring $|\psi_g\rangle$. In addition,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

we are given the ability to implement the reflection operator around the initial state $R_{\psi_0} = I - 2|\psi_0\rangle\langle\psi_0|$ and an operation that, when restricted to the subspace spanned by $\{|\psi_g\rangle, |\psi_b\rangle\}$, acts as the reflection around the target state $R_{\psi_g} = I - 2|\psi_g\rangle\langle\psi_g|$.

Then, amplitude amplification allows us to boost the success probability to 1 through repeated calls to an operator $W = -UR_{\psi_0}U^\dagger R_{\psi_g}$, from the initial state $U|\psi_0\rangle = |\psi\rangle$. The standard analysis [186] proceeds by letting $a = \sin(\theta)$ and $b = \cos(\theta)$, and showing that the 2D subspace spanned by $|\psi_g\rangle, |\psi_b\rangle$ is invariant under W, which acts as a rotation operator such that $|\psi_g\rangle\langle\psi_g|W^m|\psi\rangle = \sin((2m+1)\theta)|\psi_g\rangle$.

The algorithm can also be viewed through the lens of quantum singular value transformation (QSVT) whereby U provides a generalized block-encoding (known as a projected unitary encoding) of the amplitude a. We can see this from $|\psi_g\rangle\langle\psi_g|U|\psi_0\rangle\langle\psi_0| = a|\psi_g\rangle\langle\psi_0|$. We choose to apply a polynomial $f(\cdot)$ satisfying the quantum signal processing conditions and $f(a) = 1$ to the block-encoded amplitude [429, Theorem 27 & 28]. For example, the textbook version of amplitude amplification is recovered by setting the QSVT rotation angles to $\pm\pi/2$.[1] This QSVT circuit applies a degree $2m+1$ Chebyshev polynomial of the first kind T_{2m+1} to the amplitude a, such that $|\psi_g\rangle\langle\psi_g|W^m|\psi\rangle = T_{2m+1}(a)|\psi_g\rangle = (-1)^m \sin((2m+1)\theta)|\psi_g\rangle$ for $a = \sin(\theta)$.

Dominant resource cost (gates/qubits)

The number of calls to W is

$$m = \frac{\pi}{4\arcsin(a)} - \frac{1}{2} = O\left(a^{-1}\right)$$

for small a. Each call to W requires a call to each of $U, U^\dagger, R_{\psi_0}, R_{\psi_g}$. Often we have $|\psi_0\rangle = |0^{n+k}\rangle$, and U acts on n register qubits and k ancilla qubits such that $U|0^{n+k}\rangle = a|\psi_g\rangle_n|0^k\rangle_k + b|\perp\rangle_{n,k}$, where $|\perp\rangle_{n,k}$ denotes a state orthogonal to $|0^k\rangle$ on the ancilla register. In this case the reflection operators are simple to implement using multicontrolled Toffoli gates.

Caveats

The textbook version of amplitude amplification assumes that the success amplitude a exactly equals $\sin(\pi/(4m+2))$ for an integer m. If this is not the case (e.g., when $a = 1/\sqrt{2}$), we can introduce a new qubit in $|0\rangle$ and apply an $R_y(2\phi)$ gate (i.e., a rotation about Y by angle 2ϕ) to reduce the success probability (now defined by measuring $|\psi_g\rangle|0\rangle$) to $a\cos(\phi) = \sin(\pi/(4m'+2))$ for an integer m'.

[1] These rotation angles enable a gate compilation that removes the need for the QSVT ancilla qubit.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

In cases where we can only *lower bound* the success amplitude $a \geq a_0$, it is common to use fixed-point amplitude amplification [1067]. This is best understood through QSVT [429, Theorem 27], where the reflection operators are replaced by parameterized phase operators $e^{i\theta|\psi_g\rangle\langle\psi_g|}$ and $e^{i\phi|\psi_0\rangle\langle\psi_0|}$.[2] The QSVT rotation angles are chosen to implement a polynomial that maps *all* amplitudes taking value at least a_0 to at least $(1 - \epsilon)$. The fixed-point amplitude amplification circuit uses a QSVT circuit that makes $O(a_0^{-1} \log(\epsilon^{-1}))$ calls to $U, U^\dagger, e^{i\theta|\psi_g\rangle\langle\psi_g|}$, and $e^{i\phi|\psi_0\rangle\langle\psi_0|}$.

Example use cases

- Combinatorial optimization.
- Convex optimization via "minimum finding" subroutine (see [48, Appendix C]).
- Weakening cryptosystems.
- Tensor principal component analysis.
- Hamiltonian simulation using linear combinations of unitaries.

Further reading

- Both amplitude amplification and Grover search can be viewed through the lens of quantum walks on suitably constructed graphs. The quantum walks also take the form of a product of two reflections and more generally can be understood as quantizing a Markov chain describing a classical random walk [974]. We refer the interested reader to [276, 733, 52, 427].
- Oblivious amplitude amplification: Amplitude amplification can be extended to the case of oblivious amplitude amplification (OAA) [135]. The original formulation considered a setting where one is given unitary U such that for any state $|\psi\rangle$, we have

$$U|0^m\rangle|\psi\rangle = a|0^m\rangle V|\psi\rangle + b|0_\perp^m\phi\rangle$$

for a unitary operator V. The goal is to amplify the probability for the state $|0^m\rangle V|\psi\rangle$ to 1. This is achieved through $O(a^{-1})$ applications of an operator $W = U(I - 2|0^m\rangle\langle 0^m|)U^\dagger(I - 2|0^m\rangle\langle 0^m|)$ applied to $U|0^m\rangle|\psi\rangle$. We see that W does not require reflections around the initial state $|\psi\rangle$. We can recognize U as an m-qubit block-encoding of the operator aV, which can be transformed to a block-encoding of V using QSVT.[3] The OAA subroutine is used in

[2] It is shown in [687, Section 8.5] how these phase operators can be constructed using the corresponding controlled reflection operator. If only the uncontrolled reflection is available, a control can be added using, for example, [744, Fig. 5].

[3] We note that in this interpretation, one may be concerned that the phase information of the unitary V is lost by transforming the singular values. This turns out not to be problematic, as the phase information of V can be considered stored in the basis transformation matrices

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the context of Hamiltonian simulation via Taylor series, where it would be problematic to have to reflect around the initial state during amplification.[4] It is also used in [297] (applied to isometries) for simulation of open quantum systems. OAA requires the block-encoded operator being amplified to preserve state norms (i.e., it must be an isometry), as this ensures that the success probability of the operation is independent of the state to which it is applied, which in turn enables amplification without reflection around the initial state.

It is also possible to amplify a block-encoding of a non-isometric operator A using QSVT; see [429, Theorem 30] and [715]. Assume $\|A\| = 1$; given the ability to implement a block-encoding U of $\sqrt{p}A$, we can use oblivious amplitude amplification to implement a block-encoding of A using $O(1/\sqrt{p})$ calls to $U, U^\dagger$. Note that for a general normalized state $|\psi\rangle$, it holds that $\||A|\psi\rangle\| \leq 1$, with equality only achieved when $|\psi\rangle$ is the singular vector corresponding to the largest singular value of A. As a result, to boost the success probability of outputting $A|\psi\rangle/\||A|\psi\rangle\|$ to unity for a general input state requires using regular amplitude amplification, involving reflections around the initial state.

- While we are unaware of a standard reference for the use of an additional ancilla qubit to account for cases where the success amplitude $a \neq \sin(\pi/(4m+2))$ for integer m, discussed above in §Caveats, it is explained more fully in [754, Appendix B].

14.2 Amplitude estimation

Rough overview (in words)

Given a quantum subroutine that succeeds with unknown success probability, amplitude estimation provides quadratic speedup over classical methods for estimating the success probability. Because many quantities of interest can be encoded in an amplitude or probability, amplitude estimation can be used as a widely applicable tool for obtaining Monte Carlo estimates with complexity $O(1/\epsilon)$, instead of the $O(1/\epsilon^2)$ achieved by classical estimation.

present in the singular value decomposition, rather than in the diagonal singular values matrix. This is taken care of automatically using QSVT. Phases are preserved when using an odd polynomial.

[4] More precisely, a robust version of OAA is used which is applicable to an operator that is ϵ close to being unitary [137, 136].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Rough overview (in math)

We are given an initial state $|\psi_0\rangle$, a target ("good") state $|\psi_g\rangle$, and a unitary U and its inverse $U^\dagger$ such that

$$U|\psi_0\rangle = |\psi\rangle = a|\psi_g\rangle + b|\psi_b\rangle\,,$$

where $|\psi_b\rangle$ is a ("bad") state orthogonal to the target state. We assume that we can mark the target state $|\psi_g\rangle$ (i.e., the ability to reflect about the state). Thus, $p = |a|^2$ is the success probability of applying U and measuring $|\psi_g\rangle$.[5] We are given the ability to implement the reflection operator around the initial state $R_{\psi_0} = I - 2|\psi_0\rangle\langle\psi_0|$ and an operation that, when restricted to the subspace spanned by $\{|\psi_g\rangle, |\psi_b\rangle\}$, acts as the reflection around the target state $R_{\psi_g} = I - 2|\psi_g\rangle\langle\psi_g|$. We can then estimate the success probability by performing quantum phase estimation on an operator $W = -UR_{\psi_0}U^\dagger R_{\psi_g}$, from the initial state $U|\psi_0\rangle = |\psi\rangle$. The standard analysis [186] proceeds by letting $|a| = \sin(\theta)$ and $|b| = \cos(\theta)$ (thus, the phases of a and b are absorbed into $|\psi_g\rangle$ and $|\psi_b\rangle$ and are not determined by the following procedure) and showing that the 2D subspace spanned by $\{|\psi_g\rangle, |\psi_b\rangle\}$ is invariant under W, where it acts as a rotation operator

$$W = \begin{pmatrix} \cos(2\theta) & \sin(2\theta) \\ -\sin(2\theta) & \cos(2\theta) \end{pmatrix}.$$

This operator has eigenvalues $e^{\pm 2i\theta}$, and we can estimate θ to additive error ϵ through quantum phase estimation. The estimate for θ can be converted into an estimate for $|a|$, or for the success probability $p = |a|^2$, which is often the quantity of interest.

Dominant resource cost (gates/qubits)

The classical approach for learning the probability p to precision ϵ has complexity scaling as $M = O(1/\epsilon^2)$, where the basic idea is to perform M incoherent repetitions of applying U and measuring in the $|\psi_g\rangle, |\psi_b\rangle$ basis, and then tally the measurement outcomes and construct the frequentist (or maximum likelihood) estimate of p. Amplitude estimation provides a quadratic speedup, learning the probability (and amplitude) with complexity scaling as $M = O(1/\epsilon)$. The textbook variant has a constant success probability, which can be boosted to $1 - \delta$ with $O(\log(1/\delta))$ overhead through standard methods (e.g., probability amplification by majority voting).

[5] Note that the original paper introducing amplitude estimation [186] uses the variable a to denote the success probability. While the algorithm is referred to as amplitude estimation, it is often the success probability that we wish to compute, and the complexity of the algorithm is often presented accordingly.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

More precisely, following the analysis of [186] one can see that to learn $|a|$ to error ϵ it suffices to utilize M controlled applications of the walk operator W where M satisfies[6]

$$\epsilon \geq \frac{\pi\sqrt{1-|a|^2}}{M} + \frac{|a|\pi^2}{2M^2}\,. \tag{14.1}$$

The algorithm succeeds with probability at least $8/\pi^2$. For $|a| \approx 1 - O(\epsilon)$, a further quadratic improvement is obtained (i.e., $M = O(1/\sqrt{\epsilon})$ suffices).

To learn the success probability $p = |a|^2$ to error ϵ it suffices to utilize M controlled applications of the walk operator W where M satisfies [186]

$$\epsilon \geq \frac{2\pi\sqrt{p(1-p)}}{M} + \frac{\pi^2}{M^2}\,. \tag{14.2}$$

The algorithm once again succeeds with probability at least $8/\pi^2$. Similar to above, if $p \approx O(\epsilon)$ or $p \approx 1 - O(\epsilon)$, then it suffices to take $M = O(1/\sqrt{\epsilon})$.[7]

The overall gate complexity of an application involving amplitude estimation is given by M times the gate complexity of implementing a controlled application of W.

A common setting is the case where $|\psi_0\rangle = |0^{n+k}\rangle$, and U acts on n register qubits and k ancilla qubits such that $U|0^{n+k}\rangle = a|\psi_g\rangle|0^k\rangle_k + b|\psi_b\rangle|0^k_\perp\rangle_k$. In this case, the reflection operators are simple to implement, and W can be controlled by making these reflections controlled (adding another control qubit to a multicontrolled Z gate). We require $\log(M)$ ancilla qubits for phase estimation (which can be reduced using modern variants, see below and [855]).

Caveats

The textbook version of amplitude estimation described above produces biased estimates of $|a|$ and p. This is partly inherited from the biased nature of textbook quantum phase estimation. However, even if unbiased variants of phase estimation are used, the amplitude and probability estimates are not immediately unbiased, as they are obtained by applying nonlinear functions to the

[6] Specifically, Lemma 7 of [186] shows that if $\theta = \arcsin(|a|)$ and $\tilde{\theta} = \arcsin(|\tilde{a}|)$, then $|\theta - \tilde{\theta}| \leq \eta$ implies $|a^2 - \tilde{a}^2| \leq 2\eta\sqrt{a^2(1-a^2)} + \eta^2$. This is easily adapted to show that it also implies $|a - \tilde{a}| \leq \eta\sqrt{1-a^2} + a\eta^2/2$. They show that with probability at least $8/\pi^2$, θ is learned up to additive error at most $\eta = \pi/M$ with M calls to W, which together with the above expressions implies Eqs. (14.1) and (14.2).

[7] We can compare to the classical approach of estimating p by flipping a p-biased coin M times. Letting $\tilde{p}$ denote the estimate, which has mean p and variance $p(1-p)/M$, Chebyshev's inequality implies that $|p - \tilde{p}| \leq \epsilon$ with probability at least $8/\pi^2$ as long as $M \geq Cp(1-p)/\epsilon^2$ where $C = 1/(1 - 8/\pi^2)$. Thus, when $p \approx O(\epsilon)$ or $p \approx 1 - O(\epsilon)$, the classical approach achieves $M \sim 1/\epsilon$, and the quantum speedup is never more than quadratic.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

estimate of the phase. Unbiased variants of amplitude [855][8] and probability estimation [49, 308] have been developed to address this.

The variant of amplitude estimation described above is also "destructive" in the sense that the output state is collapsed into a state $\frac{1}{\sqrt{2}}(|\psi_g\rangle \pm i|\psi_b\rangle) \neq |\psi_0\rangle, |\psi\rangle$. A nondestructive variant may be desired if the initial state is expensive to prepare and we require coherent or incoherent repetitions of amplitude estimation. Nondestructive variants have been developed in [499, 308, 855].

Example use cases

- Approximate counting of solutions marked by an oracle (e.g., topological data analysis, combinatorial optimization).
- Amplitude estimation provides a quadratic speedup for Monte Carlo estimation [773, 642] with uses in pricing financial assets. The general idea is to prepare a state $|\psi\rangle = \sum_x \sqrt{p(x)f(x)}|x\rangle|0\rangle + |\phi 0^{\perp}\rangle$ where $\mathbb{E}[f(x)] = \sum_x p(x)f(x)$ represents the expectation value we wish to evaluate using Monte Carlo sampling and corresponds to the probability that we measure the second register in state $|0\rangle$. Hence, amplitude estimation provides a quadratic speedup for estimating this quantity.
- A special case of amplitude estimation is overlap estimation [637], where given two states $|\psi\rangle, |\psi_0\rangle$ and a unitary such that $|\psi\rangle = U|\psi_0\rangle$, the goal is to measure $\langle\psi_0|U|\psi_0\rangle = \langle\psi_0|\psi\rangle$. This can be viewed as an application of amplitude amplification, where $|\psi_g\rangle = |\psi_0\rangle$. As a result, we only require the ability to implement $R_{\psi_0} = I - 2|\psi_0\rangle\langle\psi_0|$, $U, U^{\dagger}$ (or equivalently R_{ψ_0} and R_{ψ}). Note that in overlap estimation, one additionally wants to determine the phase of a, which can be obtained by applying amplitude estimation on a controlled variant of U, as outlined in [637]. Overlap estimation can be used for estimating observables, for example, in quantum chemistry.
- A generalization of amplitude estimation, via the quantum gradient algorithm, forms a core subroutine in some approaches for quantum state tomography [49]. Pure state tomography can be thought of as a generalization of amplitude estimation, in which we seek to learn all amplitudes individually, rather than only a single aggregate quantity. Closely related work on multivariate amplitude estimation [310] has broad applicability, including in convex optimization [51] and finance [361].

[8] In order to achieve bias $\leq \epsilon\eta$, the algorithm of [855] pays a multiplicative cost overhead $\sim \frac{1}{\eta}$ which, up to logarithmic factors, could also be achieved by merely improving the precision to $\epsilon\eta$. The additive $\sim \log(\frac{1}{\epsilon\eta})$ cost overhead of [49, 308] is much more satisfactory.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Further reading

- Variants of amplitude estimation using fewer ancilla qubits (including ancilla-free approaches), or with depth-repetition tradeoffs have been proposed [460], including work to make these methods nonadaptive [1007]. For a summary of these approaches and their unification within the QSVT framework, see [855].
- There has been some work on computing, optimizing, and comparing the constant prefactor of the $M = O(1/\epsilon)$ relation using different approaches to amplitude estimation, relevant for concrete resource estimates. For example, building on the analysis of [460], the method from [855] was estimated to scale roughly as $M \approx 4.7/\epsilon$ based on numerical experiments on a range of choices for ϵ and with fixed $a = 0.5$. This was observed to be about an order of magnitude better than the textbook method from [186] described above.[9] The method from [657] furthermore showed that a comparable total query complexity could be obtained while parallelizing across multiple processors, with maximum query depth roughly $0.4/\epsilon$.

[9] Asymptotically speaking, the complexity of the methods from [855, 460] scales suboptimally, as $O(\log\log(1/\epsilon)/\epsilon)$, but the extra $\log\log(1/\epsilon)$ factor grows sufficiently slowly that for practical values of ϵ it can be bounded by a small constant.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

15

Gibbs sampling

The authors are grateful to Rolando Somma for reviewing this chapter.

Rough overview (in words)

Gibbs sampling is the task of preparing a quantum state in thermal equilibrium. This task is interesting in its own right as a means of testing the thermodynamic properties of quantum systems in a controlled way, but it is also a subroutine that is surprisingly useful within other quantum algorithms. Formally, given a Hamiltonian and a temperature, the task is to prepare the *Gibbs state* (also known as the *thermal state*) of that Hamiltonian at the associated temperature, or equivalently, to sample eigenstates of the Hamiltonian with probability proportional to their Boltzmann weights (motivating the name Gibbs *sampling*).

Physically, Gibbs sampling is routinely achieved in experiments via cooling as a manifestation of open-system thermodynamics, although theoretical understanding of such processes has been largely heuristic. Computationally, quantum Gibbs sampling is the quantum analog of the same classical task in the computational basis, often achieved by Markov chain Monte Carlo (MCMC) methods. As a representative example, the Metropolis–Hastings algorithm [512] performs local accept-reject steps to construct a Markov chain whose stationary state is the classical Gibbs distribution; the Gibbs distribution can be efficiently sampled if the Markov chain mixes rapidly. Nowadays, Monte Carlo methods have already outgrown their original intent and found ubiquitous applications in optimization and machine learning due to their simplicity and versatility. It is natural to wonder if the same features will be present for quantum Gibbs sampling.

The most direct quantum algorithms for Gibbs sampling (for noncommuting Hamiltonians) suffer from an explicit cost exponential in the size of the sys-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tem. Another approach is to quantize classical Monte Carlo algorithms [984], but this approach has faced serious technical challenges rooted in quantum mechanics: the energy-time uncertainty principle (for imposing the Boltzmann weights) and the no-cloning theorem (for "rejecting" a quantum state). Recently, a new wave [256, 939, 856, 260, 259] of proposals revisits the issue from the angle of open-system thermodynamics and gives nature-inspired algorithms for Gibbs sampling. These more directly resemble the dynamical process of thermalization and have the potential to achieve better runtimes for specific systems where thermalization is expected to be fast.

Rough overview (in math)

Given a Hamiltonian $H = \sum_i E_i |\psi_i\rangle\langle\psi_i|$ over n qubits, a desired inverse temperature β, and an error parameter ϵ, the Gibbs sampling task is to prepare an n-qubit quantum state ρ such that

$$\|\rho - \sigma_\beta\|_{\mathrm{tr}} \leq \epsilon\,,$$

where

$$\sigma_\beta := \frac{e^{-\beta H}}{\mathcal{Z}} \propto \sum_i e^{-\beta E_i} |\psi_i\rangle\langle\psi_i| \quad \text{and} \quad \mathcal{Z} := \mathrm{tr}[e^{-\beta H}].$$

The quantity $\mathcal{Z}$ is known as the partition function. The above uses the convenient error metric given by the trace norm $\|\cdot\|_{\mathrm{tr}}$, which controls the error for arbitrary bounded (possibly nonlocal) observables. In some applications, it could be sufficient to give a state ρ that approximates all *local* observables up to high precision, even if the global distance between ρ and σ_β is large. Note that σ_β corresponds to an ensemble of eigenstates of H, where an eigenstate with energy E_i occurs with probability proportional to the Boltzmann weight $e^{-\beta E_i}$ (β has units of inverse energy so that βE_i is dimensionless).

To solve this problem, the quantum algorithm requires access to H, for example, through a block-encoding of H. Block-encodings can often be efficiently constructed, for instance, when H is a sparse matrix or when H is given as a sum of poly(n) local interaction terms. Henceforth, assume that H is offset such that it is guaranteed to be a non-negative operator (no negative energies).

An early approach [838] for Gibbs sampling relied on quantum phase estimation (QPE) and amplitude amplification. In particular, one starts with a $2n$-qubit maximally entangled state (for which the reduced density matrix on the first n qubits is the maximally mixed state) and applies QPE to the first n qubits, reading an estimate of the energy into an ancilla register. Under the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

simplification that QPE has perfect resolution, one now has the state

$$\frac{1}{\sqrt{2^n}} \sum_i |\psi_i\rangle|\phi_i\rangle|E_i\rangle \,,$$

where $|\psi_i\rangle$ is the i-th eigenstate of H, E_i is the associated energy, and the states $|\phi_i\rangle$ form an arbitrary (unimportant) orthonormal basis. Next, one coherently rotates an ancilla qubit to put the correct Boltzmann weight into the amplitude:

$$\frac{1}{\sqrt{2^n}} \sum_i |\psi_i\rangle|\phi_i\rangle|E_i\rangle\left(\mathrm{e}^{-\beta E_i/2}|0\rangle + \sqrt{1-\mathrm{e}^{-\beta E_i}}|1\rangle\right).$$

Note that the probability of measuring the final qubit in $|0\rangle$ is precisely $\mathcal{Z}/2^n$. Rather than measure and postselect, one now performs amplitude amplification on the ancilla being $|0\rangle$ to produce

$$\frac{1}{\sqrt{\mathcal{Z}}} \sum_i \mathrm{e}^{-\beta E_i/2}|\psi_i\rangle|\phi_i\rangle|E_i\rangle$$

up to small error, which is a purification of the Gibbs state $\sigma_\beta = \mathcal{Z}^{-1}\sum_i \mathrm{e}^{-\beta E_i}|\psi_i\rangle\langle\psi_i|$. While QPE does not exactly produce the operation described above, a more complete analysis in [838, 272] shows the idea still works. This approach is akin to classical rejection sampling (see also [822]), where a state is chosen at random and accepted with probability $\mathrm{e}^{-\beta E_i}$, such that repeating until acceptance yields a sample from the correct distribution. Due to amplitude amplification, the quantum algorithm enjoys a quadratic speedup.

More advanced methods that have exponentially better ϵ dependence have since been developed. Reference [289] used a linear combination of unitaries approach to perform the imaginary-time evolution operator $\mathrm{e}^{-\beta H}$, again followed by amplitude amplification. Technically, that work assumed access to an operator similar to $\sqrt{H}$, but this requirement was removed in Gibbs samplers appearing in [48, 45], which employ a method for implementing smooth Hamiltonian functions. Alternatively, one can use the quantum singular value transformation along with a polynomial approximation to the function $\mathrm{e}^{-\beta(1-x)/2}$ on the interval $x \in [-1, 1]$ [431, Section 5.3] and combine this with (fixed-point) amplitude amplification [1067].

Another family of quantum algorithms is closer in spirit to classical Monte Carlo methods. They quantize the Metropolis–Hastings algorithm (quantum Metropolis sampling [984]) or simulate the dynamics arising from a system-bath interaction [256, 939, 856, 260, 259]. These algorithms make fundamental usage of QPE (or the quantum operator Fourier transform [260, 259]) for probing the energy, but most importantly (and most nontrivially), they construct

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

a detailed-balance "quantum Markov chain" via either discretely or continuously "rejecting" the quantum state. Intuitively, the algorithm implements the following rule:

"Apply a jump. If the energy increases, reject with some probability."

Care must be taken to perform the rejection step coherently and to handle the fact that the energies cannot be learned to infinite precision. Abstractly, Monte Carlo–style quantum algorithms emulate a continuous-time quantum Markov chain (i.e., a Lindbladian $\mathcal{L}$)[1] that converges to the Gibbs state after evolution time t_{mix} (often called the *mixing time*)

$$\mathcal{L}[\sigma_\beta] \approx 0 \quad \text{and} \quad \|e^{\mathcal{L}t_{\text{mix}}}[\rho_0] - \sigma_\beta\|_{\text{tr}} \leq \epsilon$$

for some initial state ρ_0. Like the classical Metropolis–Hastings algorithm, for some systems, t_{mix} can be exponentially large (or worse) in the system size n, while for other systems, it can be much smaller (polynomial or logarithmic). It is a generally difficult problem to determine t_{mix}. As a consequence, classical Monte Carlo algorithms rarely have convergence guarantees in practice. Rather, they are employed in conjunction with a variety of heuristic convergence tests.

Note that such a process can be further quantized to gain quadratic speedup [1048, 260].

Dominant resource cost (gates/qubits)

Assuming one has access to a block-encoding of the Hamiltonian H, that is, a unitary whose upper left block is the operator H/α, where α is a normalization constant at least as large as the spectral norm of H, one can accomplish the Gibbs sampling task using [48, Lemma 44] (see also [45, Corollary 16])

$$\alpha\beta\sqrt{\frac{2^n}{\mathcal{Z}}} \cdot \text{poly}(\log(1/\epsilon), n) \tag{15.1}$$

calls to the block-encoding and a similar number of other gates. Note that since we have assumed H is non-negative, we have $\mathcal{Z} \leq 2^n$. In the case that H is d-sparse, we can take $\alpha = d\|H\|_{\max}$, where $\|H\|_{\max}$ is the maximum absolute value of any entry of the matrix H. In the case one has access to $\sqrt{H}$, the β dependence can be reduced from β to $\sqrt{\beta}$ [289]. This complexity statement might be regarded as a *quadratic speedup* compared to the classical method of rejection sampling, which requires $2^n/\mathcal{Z}$ samples on average; however, note that this classical method only directly applies to diagonal (classical) Hamiltonians H.

[1] Most constructions are continuous-time quantum Markov chains as they are mathematically easier to work with than discrete-time quantum channels [984].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Otherwise, a classical approach may need to resort to exact diagonalization of H, which has $O(2^n)$ space complexity and even worse time complexity.

Monte Carlo–style quantum Gibbs sampling algorithms have complexity determined by

$$(\text{mixing time}) \cdot (\text{cost per unit evolution time } e^{\mathcal{L}}). \tag{15.2}$$

The mixing time is expected to vary significantly for different systems of interest (based on classical Monte Carlo intuition), but for systems appearing in nature, one may be optimistic based on the observed fast thermalization of physically relevant observables. In fact, in many cases, rapidly thermalizing metastable states will have physically relevant characteristics; in analogy with classical ferromagnetic systems below the critical temperature. The cost per unit evolution time is dominated by the QPE subroutine, which then scales with a certain energy resolution. Typically, we may work in a convention where H is given by poly(n) Hamiltonian terms each of strength $O(1)$, enabling a block-encoding of H with normalization $\alpha = \text{poly}(n)$ to be implemented in gate complexity poly(n), or alternatively, the ability to perform black-box Hamiltonian simulation for time t at gate cost $t \cdot \text{poly}(n)$.[2] In this case, an overall gate complexity for a rapidly mixing system (i.e., $t_{\text{mix}} \leq \text{poly}(n)$) can be poly($n, \beta, 1/\epsilon$); see [260, Table I] for a catalog of existing constructions. A recent construction [259] improved the asymptotic cost per unit Lindbladian evolution time to $\widetilde{O}(\beta)$ Hamiltonian simulation time assuming black-box access to Hamiltonian simulation; for lattice Hamiltonians, this further simplifies as one merely needs to simulate lattice patches of diameter $\widetilde{O}(\beta)$. To put together a practically relevant end-to-end resource estimate, one needs to design better algorithms (see, e.g., [351] for a single-ancilla variant for ground states) to reduce the per unit time cost, as well as to estimate the mixing time (e.g., by exact diagonalization of the map for small system sizes). Of course, if Gibbs sampling is employed as a heuristic (as in many classical applications of Monte Carlo methods), the cost will be empirical.

Caveats

On the one hand, the superpolynomial $O(\sqrt{2^n})$ complexity for Gibbs sampling that appears explicitly in Eq. (15.1) is necessary in general (for sufficiently large β it allows one to solve NP-hard or even QMA-hard problems in the general case). On the other hand, most physical Hamiltonians (if they appear to

[2] In the model of black-box Hamiltonian simulation, one can apply the unitary $U_i = e^{iHt_i}$ for user-specified choices of t_i, and the goal is to minimize the cumulative total $\sum_i t_i$ over the course of the algorithm. In this model, it is sensible for t and β to have the same units (interpreted as time), as βH and iHt are both unitless.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

thermalize in nature) should be simulable without exponential hidden prefactors. The Monte Carlo–style approach to Gibbs sampling attempts to mimic nature more closely than the other algorithms with guaranteed complexities mentioned above; hence, it looks more promising for obtaining polynomial runtimes, but this must be verified through system-specific analysis or hardware demonstrations.

Finally, if the Hamiltonian comes from classical problems (such as solving semidefinite programs), loading the instance may have exponential cost ($e^{\Omega(n)}$), which in the above presentation is hidden in the assumption of a block-encoding of classical data. Additionally, it is unclear whether Hamiltonians arising from classical data—which would likely lack the local interaction structure that one sees in chemical and physical systems—should be expected to "thermalize" quickly (i.e., whether Monte Carlo–style algorithms converge in a small number of iterations).

Example use cases

- Multiplicative weights update (MWU) method and conic programming: Gibbs sampling is the main source of quantum speedup in the MWU method, which is used to solve semidefinite programs and other conic programs [181, 182, 48, 45, 46]. Existing analyses in this direction have employed Gibbs samplers with a guaranteed quadratic (but no larger) speedup, rather than the more heuristic and recent Monte Carlo–style algorithms.
- Quantum chemistry: An important step of estimating the ground state energy of electronic structure Hamiltonians is generating an ansatz state that has a large overlap with the ground state. This might be done via Gibbs sampling at sufficiently low temperatures; the overlap with the ground state is $e^{-\beta E_0}/\mathcal{Z}$, which can be large when $1/\beta$ is sufficiently small compared to the spectral gap between the ground and excited space of the Hamiltonian.
- Condensed matter physics: Similar to quantum chemistry, Gibbs sampling provides a method for producing ansatz states for ground state energy calculations, which often capture the relevant physics. However, condensed matter physicists are also interested in material properties at finite temperatures so that the Gibbs state itself might equally be of interest.
- Computing partition functions: One of the early references to develop quantum Gibbs samplers [838] applied it to the problem of estimating the partition function $\mathcal{Z}$ up to small relative error. The partition function contains all the relevant thermodynamic information of the system.
- Combinatorial optimization: Many combinatorial optimization problems can be viewed as finding the ground state of classical Hamiltonians (i.e.,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Hamiltonians that are diagonal in the computational basis), or finding a low-energy state that achieves a high approximation ratio with the ground state. Classical Monte Carlo algorithms are a key technique in this area, and quantization of these methods can sample the same classical thermal distribution with a quadratic speedup in the mixing time [944]. The full power of Gibbs sampling for general nondiagonal Hamiltonians could be useful in situations where one adds a noncommuting transverse field to the classical Hamiltonian and wishes to prepare a low-energy state, such as in quantum annealing or for training quantum Boltzmann machines [30].

Further reading

Gibbs sampling has been studied in several specific cases. For example, [152] studied Gibbs sampling of local Hamiltonians in 1D. Moreover, [603] studied *commuting* spatially local Hamiltonians and showed conditions under which they thermalize in polynomial time, suggesting efficient Gibbs sampling via Monte Carlo–style methods. These conditions hold for any 1D system at any temperature, and in any higher-spatial dimension above a certain threshold temperature.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

16

Quantum adiabatic algorithm

The authors are grateful to Dong An for reviewing this chapter.

Rough overview (in words)

The *quantum adiabatic algorithm* (QAA) [382], sometimes referred to as *adiabatic state preparation*, is a continuous-time procedure for (approximately) preparing an eigenstate (typically the ground state) of a particular Hamiltonian of interest on a quantum device. The QAA also forms the basis for a model of quantum computation called *adiabatic quantum computation* which acts as an alternative to the standard quantum circuit model.

The main idea of the QAA is to begin in an eigenstate of a simpler Hamiltonian that is easy to prepare, and then slowly change the Hamiltonian to be equal to the more complex Hamiltonian of interest. The adiabatic theorem (see [16] and references therein), a celebrated concept from physics, dictates that if the evolution is sufficiently slow, the system will evolve to (approximately) remain in the instantaneous eigenstate of the continuously varying Hamiltonian and thus finish in the desired state. The length of time required for the evolution to succeed depends on the spectral properties of the Hamiltonian path and in particular on the minimum spectral gap. The adiabatic algorithm can be simulated on a gate-based quantum computer with time-dependent Hamiltonian simulation.

Rough overview (in math)

Let $H(s)$, where s varies as $0 \leq s \leq 1$, denote a single-parameter path through the space of Hamiltonians, and let $|\phi_j(s)\rangle$ and $E_j(s)$ denote the eigenstates and eigenvalues of $H(s)$, indexed by j in increasing order. The goal of the QAA is to prepare a certain eigenstate $|\phi_j(1)\rangle$ of $H(1)$. Let $|\psi(t)\rangle$ denote the state of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

our system at time t and let T be the total evolution time. The procedure calls for beginning in the state $|\psi(0)\rangle = |\phi_j(0)\rangle$ and allowing $|\psi(t)\rangle$ to evolve by the Schrödinger equation according to the Hamiltonian $H(t/T)$, that is, $\mathrm{i}\frac{\mathrm{d}}{\mathrm{d}t}|\psi(t)\rangle = H(t/T)|\psi(t)\rangle$ from $t = 0$ to $t = T$. Thus, as T is made larger, the path from $H(0)$ to $H(1)$ is traversed increasingly slowly.

Dominant resource cost (gates/qubits)

The main resource for the continuous-time QAA is the total evolution time T. The adiabatic theorem suggests that if T is chosen sufficiently large, and as long as eigenvalue E_j is nondegenerate along the entire path, then $|\psi(T)\rangle \approx |\phi_j(1)\rangle$ will hold. The often-quoted heuristic condition [16] for success is that

$$T \gg \max_{0\le s\le 1} \frac{\left\|\frac{dH}{ds}\right\|}{\Delta(s)^2}\,, \tag{16.1}$$

where $\Delta(s)$ is the spectral gap, that is, $\min_{i\neq j}|E_i(s) - E_j(s)|$, and $\|\cdot\|$ denotes the spectral norm. Thus, the runtime needed for the QAA to have small error is primarily governed by the minimum size of the spectral gap along the adiabatic path. This aspect of the QAA is a common sticking point as it is often difficult to produce lower bounds on $\Delta(s)$ that would suffice for proving upper bounds on T. In practice, the value of T can be chosen heuristically, or by trial-and-error, but a more detailed understanding of $\Delta(s)$ would inform smarter choices of Hamiltonian path $H(s)$.

While Eq. (16.1) is nonrigorous and potentially loose in specific scenarios, the polynomial dependence of T on the inverse spectral gap is an essential feature of the QAA. For example, it was shown in [161] that in general, any rigorous bound on the QAA runtime must scale at least linearly in the inverse spectral gap.

The QAA is typically formulated as a continuous-time procedure, but a gate-based quantum computer can simulate the QAA by discretizing the path and approximately implementing the evolution from time t to $t + \delta t$ with product formulas or with more advanced techniques for time-dependent Hamiltonian simulation. This incurs error in addition to the adiabatic error of the continuous-time QAA. The number of gates needed to do this can be made proportional to T (up to logarithmic corrections), polynomial in the number of qubits needed to hold the state $|\psi(t)\rangle$, and logarithmic in the approximation error incurred (e.g., [615]).

Caveats

A technical caveat of the QAA is that rigorous formulations of sufficient conditions for success (e.g., [567, 376]) are more complex than Eq. (16.1) and

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

likely looser than what is necessary in practice. Also, in most cases, the dependence of the runtime T on the final approximation error $\epsilon = \||\psi(T)\rangle - |\phi_j(1)\rangle\|$ goes as $T = \text{poly}(1/\epsilon)$, rather than $T = \text{polylog}(1/\epsilon)$. To circumvent this and achieve $\text{polylog}(1/\epsilon)$ dependence, one can choose more sophisticated Hamiltonian paths $H(s)$ for which all time derivatives vanish at $s = 0$ and $s = 1$ [411, 16].

A practical caveat of the QAA is that the spectral gap—the main determiner of the resource cost—is difficult to study theoretically. Numerically, it can often be computed only for small system sizes, and it is unclear whether extrapolations to larger system sizes would be accurate.

Furthermore, in many end-to-end applications the spectral gap is not only unknown, but also expected to be extremely small, implying that the QAA requires a large runtime (see §Example use cases, below). For the QAA to be efficient, the spectral gap must decay only like an inverse polynomial of the system size, but such scaling requires the problem instance to have a special structure. Such structure should not be assumed to exist without justification.

NISQ implementations

The QAA is closely related to the concept of *quantum annealing* [591], a term used especially in the context of near-term implementations on existing quantum hardware. In quantum annealing, the system is exposed to a time-dependent Hamiltonian, typically a transverse-field Ising model. The strength of the transverse field is slowly reduced, eventually to zero, where the Hamiltonian is equal to a classical Ising model encoding a hard combinatorial optimization problem. If implemented perfectly and sufficiently slowly, this would be a manifestation of the QAA, and one would obtain the solution to the problem. However, the typical setting of quantum annealing is to consider faster implementations, and to possibly allow for some amount of control noise and finite-temperature effects (rather than evolving under a closed system at zero temperature), which induce transitions from the ground state to excited states. The goal is relaxed from ending in the exact ground state of the final Hamiltonian to ending in a low-energy state that can be considered an *approximately* optimal solution to the problem. The success metric is often the quality of the solution produced rather than the runtime required to find the best solution. As such, it is a heuristic algorithm and must be compared with classical heuristic algorithms, where evidence of a scalable advantage is mixed. See, for example, [323] for a perspective on quantum annealing and the most promising related directions.

Separately, the QAA can be related to variational quantum algorithms, which are NISQ friendly. In particular, by applying product formulas to the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

QAA, one obtains alternating time evolutions by $H(0)$ and by $H(1)$; in the case that $H(0)$ is a transverse field and $H(1)$ is a classical cost function, this is precisely an instance of the quantum approximate optimization algorithm (QAOA) [384], a leading NISQ algorithm. In the limit of large depth, the QAOA can fully simulate the QAA to arbitrarily small precision. However, in a NISQ setting, the depth of the QAOA would need to be restricted, and the QAOA would not exactly follow the QAA.

Example use cases

- Combinatorial optimization: The QAA was first invented [382] as a way to solve hard classical combinatorial optimization problems on a quantum computer. An example is constraint satisfaction problems, where one is given a Hamiltonian $H(1)$ that is diagonal in the computational basis (i.e., "classical") and equal to the sum of various constraints on n bits. The ground state of $H(1)$ is the bit string that violates the fewest constraints. One typically chooses the initial Hamiltonian to be $H(0) = -\sum_{i=1}^{n} X_i$, where X_i denotes the Pauli-X operator on qubit i, whose ground state is an easy-to-prepare product state. The QAA is guaranteed to find the ground state of $H(1)$ if it is run with sufficiently large evolution time. However, in general, it is expected that the spectral gaps along the adiabatic path become exponentially small in n [638, 1071, 517, 23, 1032], indicating that the QAA requires exponentially long runtime.
- Quantum chemistry and condensed matter physics: A central problem of quantum chemistry and computational condensed matter physics is the problem of finding the ground state energy of a molecule, material, or lattice model. This can be solved efficiently with quantum phase estimation (QPE) so long as one can prepare a state that has substantial overlap with the ground state of the Hamiltonian. Adiabatic state preparation has been proposed as a method for producing such a state (see, e.g., [1056, 870, 1006, 644, 966, 1031, 1072]). This initial state preparation is often the bottleneck in the end-to-end quantum solution, as it can require exponential time for systems of interest (see, e.g., [670]).
- Quantum linear system solvers: The state-of-the-art quantum linear system solvers [313] leverage the QAA to produce a quantum state $|x\rangle$ corresponding to the solution of a linear system $Ax = b$ (see also [964, 31, 689, 572]). In particular, this method allows the runtime to scale linearly in the condition number of the matrix A.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Further reading

- See [16] for a comprehensive 2018 review of the QAA and adiabatic quantum computation more generally.
- See [162] for a digital version of the QAA for a gate-based quantum computer, but distinct from a direct simulation of the QAA. The idea is to choose a sequence of s values $0 = s_0 < s_1 < s_2 < \cdots < s_T = 1$ and perform measurements of $H(s_t)$ for $t = 0, \ldots, T$ in sequence using QPE. As long as the difference between consecutive values of s is sufficiently small, the quantum Zeno effect guarantees that each measurement will project onto the correct eigenstate $|\phi_j(s_t)\rangle$ with high probability (see also [944, 673]). One can also take larger jumps, and amplify their success probability with fixed-point amplitude amplification. The resource cost has a similar dependence on the spectral gap as the continuous-time QAA: if the "path length" traced by the eigenstate $|\phi_j(s)\rangle$ is L, the minimum gap is Δ, and the target error is ϵ, then the gate cost of the algorithm is $O(L \log(L/\epsilon)/\Delta)$. The path length L can be upper bounded by $\|dH/ds\|/\Delta$, which roughly recovers Eq. (16.1).
- Along the lines of the previous bullet, [1012] gives an alternative way to effect adiabatic state preparation on a gate-based computer with polylog$(1/\epsilon)$ overall error dependence, via quasi-adiabatic continuation.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

17

Loading classical data

The end-to-end quantum applications covered in this book have classical inputs and classical outputs, in the sense that the problem is specified by some set of classical data, and the solution to the problem should be a different set of classical data. In some cases, the input data is relatively small, and loading it into the algorithm does not contribute significantly to the cost of the algorithm. In other cases—for example, "big data" problems within the areas of machine learning and finance—the dominant costs, both for classical and quantum algorithms, can be related to how the algorithms load and manipulate this large quantity of input data. Consequently, the availability of quantum speedups for these problems is often dependent on the ability to quickly and coherently access this data. The true cost of this access is the source of significant subtlety in many end-to-end quantum algorithms.

The authors are grateful to Thomas Häner, Damian Steiger, and Xiao Yuan for reviewing this chapter.

17.1 Quantum random access memory

Rough overview (in words)

Quantum random access memory (QRAM) is a construction that enables coherent access to a database, such that multiple different elements can be read in superposition. The ability to rapidly access large, unstructured datasets in this way is crucial to the speedups of certain quantum algorithms, for example in quantum machine learning based on quantum linear algebra. QRAM is commonly invoked to circumvent data-input bottlenecks [1] in situations where loading input data could dominate the end-to-end runtime of an algorithm. It

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

remains an open question, however, whether a large-scale dedicated QRAM device will ever be practical, casting doubt on quantum speedups that rely on QRAM. Note that, while here we focus on the more common use case of loading *classical* data with QRAM, certain QRAM architectures can be adapted to also load *quantum* data [427].

Rough overview (in math)

Consider a length-N, unstructured classical data vector x, and denote the i-th entry as x_i. Let the number of bits of x_i be denoted by d. Given an input quantum state $|\psi\rangle = \sum_{i=0}^{N-1} \sum_{b\in\{0,1\}^d} \alpha_{ib}|i\rangle|b\rangle$, QRAM is defined [434] as a unitary operation Q with the action

$$Q|\psi\rangle = Q \sum_{i=0}^{N-1} \sum_{b\in\{0,1\}^d} \alpha_{ib}|i\rangle|b\rangle = \sum_{i=0}^{N-1} \sum_{b\in\{0,1\}^d} \alpha_{ib}|i\rangle|b \oplus x_i\rangle. \tag{17.1}$$

Here, the first $\log_2(N)$-qubit register stores the "address" (assuming for simplicity that N is a power of 2), while the second d-qubit register stores the corresponding "data." Note that the unitary Q can also be understood as an oracle (or black box) providing access to x, as $Q(\sum_i \alpha_i|i\rangle|0\rangle) = \sum_i \alpha_i|i\rangle|x_i\rangle$.

Let T_Q denote the "cost" of implementing the operation Q, where T_Q can be measured in wall time, circuit depth, total number of gates, total circuit spacetime occupancy, total number of T gates, etc., depending on the context. Algorithms that rely on QRAM to claim exponential speedups over their classical counterparts frequently assume that $T_Q = \text{polylog}(N)$. However, as discussed in §Caveats, below, it is crucial to emphasize that this assumption can only hold when T_Q is interpreted as the circuit depth or wall time (or something similar) to implement Q; whereas, if T_Q is taken to be the total gate cost or the spacetime occupied by the computation, simple gate counting arguments imply a lower bound of $T_Q \geq \Omega(dN)$. In a discrete gate set, each unit of spacetime can be occupied with only a finite number of unique gates; since there are 2^{dN} different possible data vectors x, the circuit must have at least $\Omega(dN)$ spacetime to be able to implement all possibilities (see also [568, Section V] for a more detailed discussion).

Dominant resource cost (gates/qubits)

Let us consider for simplicity the $d = 1$ case, that is, when each data entry is a single bit. The QRAM operation Q can be implemented as a quantum circuit that uses $O(N)$ gates. Assuming gates acting on disjoint qubits can be parallelized, a circuit depth of $T_Q = O(\log(N))$ can be achieved at the expense of using $O(N)$ ancillary qubits; explicit circuits can be found in, for example, [349, 496]. The number of ancillary qubits can be traded off for increased

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

circuit depth; circuits implementing Q can be constructed using $O(N/M)$ ancillary qubits and depth $O(M \log(N))$, where $M \in [1, N]$; see examples in [722, 140, 349, 496] (the setting of $M = N/\log(N)$ is sometimes referred to as "QROM"—see terminology caveats below—and its fault-tolerant cost of implementation is well established [75]).

If the data vector x is sparse—that is, only s of its N entries are nonzero—then there exist circuit implementations with depth as shallow as $T_Q = O(\log(s \log(N)))$, using $O(s \log(s) \log(N))$ ancillary qubits, both of which for $s = O(1)$ are exponentially better than the general case of a dense vector x [1085].

Each of the above constructions can be generalized to the $d > 1$ case with various space-time tradeoffs. For example, the d bits of a data entry can be queried in series, requiring $O(N)$ ancillary qubits with depth $T_Q = O(d \log(N))$ (improvement to $T_Q = O(d + \log(N))$ is possible for certain QRAM architectures [269]). Alternatively, the d bits can be accessed in parallel, with depth $T_Q = O(\log(N))$, but at the price of $O(Nd)$ ancillary qubits.

Caveats

The main concern for QRAM's practicality is the large hardware overhead that is necessary to realize fast queries with depth $T_Q = O(\log(N))$. This cost is likely to be prohibitive for big-data applications where N can be millions or billions. The cost will also be magnified by additional overhead associated with error correction and fault tolerance [349], especially considering that circuits implementing Q are composed of $O(N)$ non-Clifford gates. Indeed, this observation together with the assumption that magic state distillation is expensive to run in a massively parallel fashion, has led some to argue that $T_Q = O(\log(N))$ is not realistic in a fault-tolerant setting (see, e.g., [568]). However, it is possible that alternative approaches to fault tolerance tailored to QRAM could help alleviate this large hardware overhead.

The fault-tolerance overhead may be reduced for the so-called bucket-brigade QRAM (BBQRAM) [434, 62, 496], which is a family of circuits implementing Q that are intrinsically resilient to noise. More precisely, [496] shows that if ϵ is the per-gate error rate, BBQRAM circuits can implement Q with leading-order fidelity $F \sim 1 - \epsilon\,\mathrm{polylog}(N)$, while generic circuits implementing Q have leading-order fidelity $F \sim 1 - \epsilon\, O(N)$. Nevertheless, at the scale necessary for useful end-to-end applications, some amount of error correction will almost certainly be required even for BBQRAM circuits.

Even if depth $T_Q = \mathrm{polylog}(N)$ is practically achievable, some have argued that any fair comparison with state-of-the-art classical methods should then allow for classical parallel computation. After all, the parallel classical

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

hardware necessary to operate the circuit Q (including the quantum error correction) could in principle be repurposed directly toward solving the end-to-end computational problem. For example, many linear algebra tasks such as matrix-matrix and matrix-vector multiplication of size-N objects are amenable to parallelization, and can also have cost scaling as polylog(N) in some parallel classical models of computation [954, 568]. Under such a comparison, it becomes difficult to identify conditions where QRAM-based quantum algorithms can give rise to a significant scaling advantage [568].

Some terminology caveats:

- The unitary Q in Eq. (17.1) is referred to by some as quantum read-only memory (QROM) [75], reflecting the fact that Q corresponds only to reading data. Some algorithms also require the ability to write to the vector x during computation, but the writing of classical (i.e., not in superposition) data need not be implemented via a quantum circuit.
- The term QRAM is used by different authors to refer to the unitary Q, families of circuits that implement Q, or quantum hardware that runs said circuits.
- The terms QRAM and QROM are sometimes used for distinguishing the cases of $T_Q = \text{polylog}(N)$ and $T_Q = \text{poly}(N)$, respectively, even though T_Q is unrelated to the distinction between reading and writing. The term QROAM has also been used to describe intermediate circuits that trade off depth and width [140].
- Some use the term QRAM to refer exclusively to the case $N \gg 1$ and $T_Q = \text{polylog}(N)$ depth, where the implementation challenges for QRAM are most pronounced.

Elsewhere in this book, we follow the convention described in the final bullet point above: usage of the term QRAM, unless specified otherwise, refers to the ability to implement Q at cost polylog(N).

Example use cases

- Quantum linear algebra: QRAM can be used as an oracle for implementing linear algebra algorithms operating on unstructured data (e.g., by acting as a subroutine in a block-encoding), with applications in machine learning, finance, etc. For example, the quantum recommendation systems algorithm [608] (now dequantized [976]) uses QRAM as a subroutine to efficiently encode rows of an input data matrix in the amplitudes of quantum states (see Appendix A of [608] for details).
- Hamiltonian simulation, quantum chemistry, condensed matter physics: In the linear combination of unitaries input model, QRAM can be used as a

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

subroutine for "PREPARE" oracles that encode coefficients of the simulated Hamiltonian into the amplitudes of quantum states [75]. These use cases typically consider the hybrid QROM/QRAM constructions with $O(K \log(N))$ ancillary qubits and depth $O(N/K)$ (with the parameter K to be optimized), because the amount of data (and thus the size of N) scales only polynomially with the system size.

- Grover search: QRAM can be used as an implementation for Grover's oracle in the context of an unstructured database search; see Chapter 4 of [801]. This appears for example in quantum algorithms that utilize dynamic programming to give polynomial speedups for combinatorial optimization problems like the traveling salesperson problem [28]. However, it has been argued that a quantum computer running Grover's algorithm with a QRAM-based oracle would not provide a speedup over a classical computer with comparable hardware resources [954].
- Topological data analysis (TDA): A small QRAM (i.e., not exponentially larger than the main quantum data register) is used in some quantum algorithms for TDA [709, 755] in order to load the positions of the data points for computing whether simplices are present in the complex at a given length scale.

Further reading

- Reference [568] focuses on various fundamental and practical concerns for large-scale QRAM, while also providing a comprehensive survey.
- Reference [293] provides an overview of practical concerns facing QRAM in the context of big-data applications (though the discussions of noise resilience there and in [62] are somewhat outdated, cf. [496]).

17.2 Preparing quantum states from classical data

Rough overview (in words)

An important subroutine in many quantum algorithms is preparing a quantum state given a list of its amplitudes stored, for example, in a classical database.[1] The upshot is that N amplitudes, which require $O(N)$ classical bits to write down, can be encoded in a quantum state with only $\log_2(N)$ qubits, an exponential compression in memory. However, there are caveats; for example, simple information-theoretic bounds [835] dictate that the quantum circuit that

[1] When the amplitudes are given by some well-behaved function, rather than being arbitrarily chosen, different (related) protocols are used; see §Further reading, below.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

prepares the $\log_2(N)$-qubit state must still have at least $O(N)$ gates, so no exponential advantage in gate complexity is possible. Additionally, reading out the N amplitudes from the encoded quantum state generally requires full quantum state tomography, requiring $\Omega(N)$ preparations of the state. Depending on which resource is being optimized, the best protocol for state preparation will look different, and optimal state preparation methods are known for several natural choices of metric.

Rough overview (in math)

Let $x = (x_0, \ldots, x_{N-1}) \in \mathbb{C}^N$ be a vector of N complex numbers, where N is a power of 2, and let

$$|\psi\rangle = \frac{1}{||x||} \sum_{i=0}^{N-1} x_i |i\rangle \tag{17.2}$$

be the associated normalized quantum state, where $||x||$ denotes the standard Euclidean vector norm. Let $n = \log_2(N)$ denote the number of qubits of $|\psi\rangle$. The goal is to prepare the state $|\psi\rangle$ by applying a quantum circuit to the initial state $|0\rangle^{\otimes n}$. This problem has been extensively studied in the literature; a common approach, originating in [463], is to iterate through each of the n qubits and perform a single-qubit rotation, with the angle of rotation depending on the setting of the previous qubits. The rotation on the first qubit creates the 1-qubit state

$$\left(\sqrt{\sum_{i=0}^{N/2-1} |x_i|^2} \right) |0\rangle + \left(\sqrt{\sum_{i=N/2}^{N-1} |x_i|^2} \right) |1\rangle$$

by performing a single-qubit rotation (about the Y axis) on the state $|0\rangle$ by an appropriate angle. Next, a similar kind of single-qubit rotation is performed on the second qubit, where the angle of rotation depends on whether the first qubit is $|0\rangle$ or $|1\rangle$. The $(m+1)$st rotation is by one of 2^m angles, depending on the setting of the first m qubits. Thus, in total there are $1 + 2 + \cdots + 2^{n-1} = N - 1$ total angles that might be used for single-qubit rotations. This sequence of operations prepares the state $||x||^{-1} \sum_{i=0}^{N-1} |x_i| |i\rangle$. To apply the phases, a single- (or zero-) qubit phase gate with the appropriate phase "angle" is performed—the angle depends on the setting of all n qubits, corresponding to the $N = 2^n$ different phases that might be needed. Thus, the total number of angles that define the protocol is $2N - 1$, exactly corresponding to the number of real parameters needed to describe the general state in Eq. (17.2).

It remains to describe how the controlled single-qubit rotations are performed when there are many control bits and different angles for each setting of the control. Here, one has many choices and the exact method will depend

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

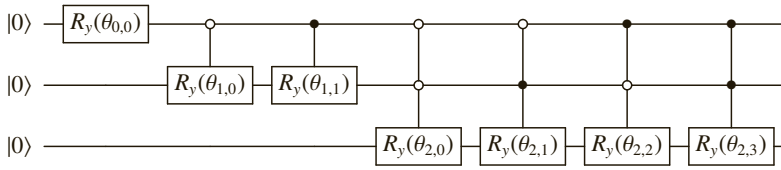


Figure 17.1 A simple quantum circuit to prepare an arbitrary state $|\psi\rangle$ with non-negative real amplitudes on $n = 3$ qubits. The gate $R_y(\theta)$ denotes a single-qubit rotation by angle θ about the Y axis. The $2^n - 1$ angles $\theta_{s,p}$ indexed by $s \in \{0, 1, \ldots, n-1\}$ and $p \in \{0, 1, \ldots, 2^s - 1\}$ can be calculated from the amplitudes x_i. To account for negative or complex amplitudes, 2^n additional controlled phase gates would be needed. More sophisticated proposals can reduce the depth for ancilla-free constructions from $O(2^n)$ to $O(2^n/n)$ [968].

on how one has access to the data in x and what resource is being optimized. The most straightforward way is to iterate through each possible setting of the control bits and perform a multiply controlled rotation by a fixed angle for each in sequence. This approach requires $O(N)$ n-qubit gates applied sequentially, as depicted in Fig. 17.1. Assuming one can perform arbitrary single-qubit gates to exact precision, it is possible to prepare the state $|\psi\rangle$ exactly. However, it is often useful to work with finite precision angles stored in binary, moreover one often needs to design circuits from a discrete gate set, such as the Clifford + T gate set, when compiling into a gate sequence that can be implemented fault tolerantly. When this is the case, single-qubit rotations must be performed approximately: to approximate a single-qubit rotation to error ϵ, a Clifford + T sequence of length $O(\log(1/\epsilon))$ can be applied [884].

When ancilla qubits are available, one can design protocols that have shallower depth (but about the same total number of gates). For example, one can store the $2N - 1$ angles in a quantum random access memory (QRAM) data structure. In this case, for $m = 1, \ldots, n$, to perform the required controlled rotations on the mth qubit in superposition, one can (i) read in the (approximate) binary value of the rotation angle (depending on the setting of the first $m - 1$ qubits) into an ancilla register using a single query to the QRAM, and then (ii) perform fixed-angle single-qubit rotations on the mth qubit, controlled by the qubits of the ancilla register storing the binary representation of the rotation angle, and finally (iii) reset (uncompute) the ancilla register with another query to the QRAM. This way, one applies the correct angle in parallel, rather than iterating through all possible 2^{m-1} angles.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost (gates/qubits)

In Table 17.1, we collect several state preparation results in the model where any single-qubit gate can be performed exactly and the only multiqubit gates allowed are CNOTs. The circuit size (i.e., the total number of single-qubit and CNOT gates) and depth (i.e., the number of parallel-acting layers of gates), as well as the number of ancilla qubits (i.e., the number of qubits beyond the n qubits needed to hold the state $|\psi\rangle$) are listed.

Ref.	Circuit size	Circuit depth	Ancilla qubits
[968, 1073]	$O(2^n)$	$O(2^n/n)$	none
[968, 1073]	$O(2^n)$	$O(2^n/(m+n))$	$m \in [0, O(2^n/n)]$
[968, 1085, 468]	$O(2^n)$	$O(n)$	$O(2^n)$

Table 17.1 Asymptotic resource cost (and tradeoffs therein) of exact state preparation of abitrary states in a gate set with CNOT gates and arbitrary single-qubit gates.

Note that the result of [1073], which shows depth $O(2^n/(m+n))$ using m ancilla qubits for $m \leq O(2^n/n)$, encompasses all other results in the table (and is superior to the third row as it uses $O(2^n/n)$ ancilla qubits instead of $O(2^n)$). We include the other results for completeness, as they are distinct constructions and can have other potential upsides.

A lower bound of $\Omega(2^n)$ is known for circuit size [835], so all of the results above are size optimal up to constant factors. Moreover, for any m ancillas, a lower bound of $\Omega(\max(n, 2^n/(n+m)))$ is known for the circuit depth [968], so all of the results above are also optimal in circuit depth, up to constant factors.

For approximate state preparation using a discrete gate set such as Clifford + T, the state $|\psi\rangle$ is prepared up to ϵ error in ℓ_2-norm, so that the circuit size and depth depends on ϵ. Results in this model are collected in Table 17.2.

Ref.	Circuit size	Circuit depth	Ancilla qubits
[968]	$O(2^n \log(2^n/\epsilon))$	$O\left(\frac{2^n}{n} \log(2^n/\epsilon)\right)$	none
[968]	$O(2^n \log(2^n/\epsilon))$	$O\left(\frac{2^n}{m+n} \log(2^n/\epsilon)\right)$	$m \in [0, O\left(\frac{2^n}{n \log(n)}\right)]$
[1084]	$O(2^n \log(1/\epsilon))$	$O\left(\frac{2^n}{m} \log(m) \log(\log(m)/\epsilon)\right)$	$m \in [0, O(2^n)]$
[468]	$O(2^n \log(n/\epsilon))$	$O(n + \log(1/\epsilon))$	$O(2^n)$

Table 17.2 Asymptotic resource cost (and tradeoffs therein) of approximate state preparation using the Clifford + T gate set.

If the state $|\psi\rangle$ is sparse, meaning that only s of the N amplitudes are nonzero, then more efficient state preparation methods are known. In particular, [1085, 968, 727] have studied shallow-depth circuits for sparse state

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

preparation, achieving circuit depth $O(\log(ns))$ using only $O(ns/\log(s))$ ancilla qubits [727], a great improvement over the general case when $s \ll N$.

In some fault-tolerant implementation schemes, such as lattice surgery using surface codes, Clifford gates can be performed cheaply, while T gates require the expensive process of magic state distillation. While $\Omega(2^n \log(1/\epsilon)/\log(n))$ total gates are necessary [801, Eq. 4.85] to approximately create $|\psi\rangle$, [722] noted that it is possible to reduce the number of T gates to $\sqrt{2^n}\log(2^n/\epsilon)$ using $\sqrt{2^n}\log(1/\epsilon)$ ancillas (in fact, there is a smooth tradeoff between the T count and the number of ancillas). Furthermore, these ancillas can be *dirty*, meaning that they can be initialized into any quantum state and they are returned to their (potentially unknown) initial state at the end of the procedure.

All of the above constructions are "garbage-free" state preparation protocols, because they prepare the state $|\psi\rangle$ exactly and all ancilla qubits are returned to their initial state. However, in some applications, it is allowed to leave an ancilla register entangled with the data as long as the amplitudes are correct. That is, one might prepare the state

$$\frac{1}{\|x\|}\sum_{i=0}^{N-1} x_i|i\rangle \otimes |\text{garbage}_i\rangle\,.$$

In this setting, en route to giving better algorithms for the electronic structure problem, [75, Section IIID] gave a construction that approximately prepares the above state using only $O(N + \log(1/\epsilon))$ T gates, albeit still requiring $O(N\log(1/\epsilon))$ Clifford gates and $O(\log(N/\epsilon))$ ancillas. In [75], the construction is presented with $O(N)$ depth, but it could be improved to $O(\log(N))$ depth at the expense of additional ancillas, using log-depth constructions for QRAM, and it could also be combined with the space-time tradeoffs mentioned above, as discussed in [722, 140].

Caveats

- Classical preprocessing: Computing the circuits for preparing $|\psi\rangle$ given the list of N coefficients x can be a non-negligible classical cost. For example, computing each of the $O(N)$ single-qubit rotation angles requires computing sums and evaluating trigonometric functions, which can be done to precision ϵ in $\text{polylog}(1/\epsilon)$ classical time. Moreover, computing Clifford + T gate sequences that approximate given rotation angles to error ϵ likewise requires $\text{polylog}(1/\epsilon)$ classical time [884]. The total classical work scales as $O(N\,\text{polylog}(1/\epsilon))$, although this cost can be parallelized.
- Coherent arithmetic: To avoid some of the classical preprocessing, one might try to perform the arithmetic coherently. This might be unavoidable if the entries of x arrive in an online fashion and rotation angles and other

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

quantities need to be computed after superpositions have been created. Formally, the scaling of coherent arithmetic is mild, generally requiring just polylog$(N, 1/\epsilon)$ number of gates and ancilla qubits, but in practice this is likely to be expensive. For example, known methods for coherently computing arcsin$(\cdot)$ to nine bits of precision use order-10^4 Toffoli gates and more than 100 ancilla qubits [494]. See [894] for a general black-box approach that avoids coherent arithmetic.

- Too many ancilla qubits: Achieving depths that scale logarithmically with N requires $O(N)$ ancilla qubits, which limits the size of N that might be practical. This could be mitigated if it is possible to develop a large-scale hardware element specialized for performing the sort of circuits that arise in these protocols, similar to a QRAM.
- Long-range gates: Achieving polylog(N) depth for state preparation requires $O(N)$ ancilla qubits and $O(N)$ gates, many of which act in parallel and on far-separated qubits. If spatial locality were imposed, it would likely be difficult to avoid $O(N)$ overhead in depth.
- Dequantization: Consider the task of drawing samples from the same probability distribution induced by measuring $|\psi\rangle$ in the computational basis in time polylog(N) time. Preparing $|\psi\rangle$ as described is a quantum method of doing so, but the same can be done classically by first constructing a certain classical data structure and assuming access to classical RAM [248]. In some machine learning applications, this idea leads to classical algorithms that effectively dequantize quantum algorithms that utilize the state preparation primitive [976, 977].

Example use cases

- Hamiltonian simulation via linear combination of unitaries (LCU) requires a PREPARE step where a state is prepared with certain classically computed coefficients. Relatedly, the same PREPARE gadget is used to construct block-encodings of such Hamiltonians. However, in this application, state preparation with garbage is generally allowable.
- In certain quantum machine learning protocols, classical data (e.g., image pixel values) are encoded into a quantum state via the so-called "amplitude encoding," where N classical features are stored in a quantum state of $\log_2(N)$ qubits [916]. Following the preparation of the amplitude-encoded data, the state is processed with the goal of, for example, classifying the image.
- Creating a block-encoding of a matrix of classical data is performed using state preparation as a subroutine (more precisely, block-encoding classical data requires controlled state preparation). The block-encoding is then use-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ful in a variety of contexts, for example in quantum interior point methods, and certain quantum machine learning algorithms.

Further reading

- When the amplitudes x_i correspond to an efficiently computable function $f(i)$, the complexity of state preparation can be reduced. In this case, the oracle access to x_i can be replaced by a reversible computation of $f(i)$, up to t bits of precision, using coherent arithmetic $|i\rangle|0^t\rangle \to |i\rangle|f(i)\rangle$ [494, 149, 791]. The value of $f(i)$ can be *transduced* into the amplitude using the methods of [465, 894, 1020, 106], and the success probability boosted to unity using quantum amplitude amplification. There is an alternative method [754], based on quantum singular value transformation (QSVT) that circumvents the need for the coherent evaluation of $f(i)$ by implementing a low-cost block-encoding of $\sin(i)$, and then using QSVT to apply $f(\arcsin(\cdot))$ to this block-encoding. The complexity of both of these approaches depends on an "ℓ_2-norm filling-fraction" $\mathcal{F}_f^{[N]} := \|f(i)\|_2/(\sqrt{N}|f(i)|_{\max})$ as $O(1/\mathcal{F}_f^{[N]})$ (see [754] for more detail). There is also an approach [859] based on the adiabatic algorithm which has a worse dependence on $\mathcal{F}_f^{[N]}$. For efficiently integrable probability distributions, one can use the approach of [463], which has complexity independent of $\mathcal{F}_f^{[N]}$. However, this approach requires coherent arithmetic to reversibly evaluate the integral of the desired function (when applied to functions for which an analytic expression for the integral is not available, this can nullify the quadratic speedup in quantum-accelerated Monte Carlo estimation [521]). There also exist methods specialized for certain target states, such as Gaussians [623, 860].
- A related problem asks to synthesize an arbitrary $2^n \times 2^n$ unitary. Without ancillas, this requires depth and size $O(4^n)$, for which there are upper [787] and lower [931] bounds that match up to constant factors. With ancillas, it is an open question whether or not the depth can be reduced to poly(n); this is related to the "unitary synthesis problem" from the list of open problems in [3], and it has been studied in several works, for example, [968, 883, 1073]. A depth lower bound of $\Omega(n + 4^n/(m + n))$ is known for m ancilla qubits [968], but the shallowest upper bound is depth $O(n2^{n/2})$, using $m = O(4^n/n)$ ancilla qubits [1073].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

17.3 Block-encoding dense matrices of classical data

Rough overview (in words)

Many potential applications of quantum algorithms, especially in the area of machine learning, require access to large amounts of classical data, and in order to process this data on quantum devices, one needs coherent query access to the data. Block-encoding is a technique for importing classical data into quantum computers that provides exactly this type of coherent query access. Block-encodings work by encoding the matrices of classical data as blocks within larger matrices, which are defined such that the full encoding is a unitary operator. One way of thinking of this process is by "brute-force" compiling a unitary with the right structure, and then postselecting measurement outcomes to ensure the desired block of the unitary was applied. In general, block-encoding a dense matrix is not an efficient process, as both the normalization factor of the block-encoding and the circuit complexity to implement it can scale with the size of the matrix (e.g., poly(N) for an $N \times N$ matrix). Nonetheless, end-to-end applications often assign *polylogarithmic* cost to the block-encoding, which is achievable if the relevant cost metric is the circuit depth (rather than the circuit size)—this is similar to the assumption that one has access to large-scale log-depth quantum random access memory (QRAM). For a general treatment not restricted to dense classical data, see Section 10.1 on block-encoding.

Rough overview (in math)

Given an $N \times N$ matrix A, a block-encoding is a way of encoding the matrix A as a block in a larger unitary matrix:

$$U_A = \begin{pmatrix} A/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix}.$$

If A is not square, one can pad it with zeros such that it becomes square. Let $n = \lceil \log_2(N) \rceil$. We say that the $(n + a)$-qubit unitary U_A is an (α, a, ϵ)-block-encoding of the matrix $A \in \mathbb{C}^{N \times N}$ if

$$\left\| A - \alpha(\langle 0|^{\otimes a} \otimes I) U_A (|0\rangle^{\otimes a} \otimes I) \right\| \leq \epsilon,$$

where $a \in \mathbb{N}$ represents the number of ancilla qubits needed, $\alpha \in \mathbb{R}_+$ is a normalization constant, and $\epsilon \in \mathbb{R}_+$ is an error parameter. The fact that U_A is unitary implies that the normalization constant α must satisfy $\alpha \geq \|A\|$, where $\|\cdot\|$ denotes the spectral norm.

In this section, we consider the case where the N^2 entries of A are arbitrary values provided to us in a classical database. In general, all of these entries

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

can be nonzero; that is, A is a dense matrix. The goal is to provide a quantum circuit implementing a unitary U_A as above while minimizing quantities such as the circuit depth and circuit size, as well as the number of ancilla qubits a and the normalization constant α. The block-encoding construction we focus on performs U_A using a pair of state preparation unitaries [429, 608, 248], following the more general method of block-encoding Gram matrices [429, Lemma 47]. In particular, the product

$$U_A = U_R^\dagger U_L$$

is an exact $(\alpha, a, 0)$-block-encoding of A, where U_L and U_R are unitaries that perform (controlled) state preparation. Specifically, the $(n+a)$-qubit unitaries U_L and U_R prepare a different $2n$-qubit state for each of 2^n possible settings of the final n-qubit register, with the assistance of $a-n$ additional ancilla qubits, as follows:

$$\begin{aligned} U_L|0\rangle^{\otimes(a-n)}|0\rangle^{\otimes n}|i\rangle &= |0\rangle^{\otimes(a-n)}|\psi_i\rangle \\ U_R|0\rangle^{\otimes(a-n)}|0\rangle^{\otimes n}|j\rangle &= |0\rangle^{\otimes(a-n)}|\phi_j\rangle\,, \end{aligned} \tag{17.3}$$

where the $2n$-qubit states $|\psi_i\rangle$ and $|\phi_j\rangle$ are chosen such that $\langle\psi_i|\phi_j\rangle = A_{ij}/\alpha$, where A_{ij} is the matrix entry of A in row i and column j—the states $|\psi_i\rangle$ and $|\phi_j\rangle$ encode the (normalized) rows of A and norms of those rows, respectively. For this construction, the normalization constant α satisfies

$$\alpha = \|A\|_F\,,$$

where $\|\cdot\|_F$ is the Frobenius norm. Note that for an $N \times N$ matrix the Frobenius norm satisfies $\|A\| \leq \|A\|_F \leq \sqrt{N}\|A\|$—thus, the value of α achieved by this method can be larger than its minimal possible value ($\|A\|$) by a factor as large as $\sqrt{2^n}$.[2]

There are several methods of implementing the (controlled) state preparation unitaries U_L and U_R, offering tradeoffs between various metrics, as discussed in Section 17.2 on state preparation. Of particular relevance is the T-count and T-depth of the circuit when it is decomposed into a Clifford + T gate set, as the T gate is the most difficult to implement in many fault-tolerant schemes. A general strategy for implementing U_L and U_R involves constructing binary trees representing the amplitudes in the states $|\psi_i\rangle$ and $|\phi_j\rangle$ in Eq. (17.3), and building the state preparation unitaries out of controlled Y rotations by angles

[2] See [429, Lemma 50] for a variant of this method yielding normalization factor $\alpha = \sqrt{n_q(A)n_{2-q}(A^\dagger)}$ for $q \in [0,2]$, where $n_q(A) = \max_i \|A_{i,\cdot}\|_q^q$, with $\|\cdot\|_q$ the vector q-norm and $A_{i,\cdot}$ the i-th row of A.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

	Optimized for min depth	Optimized for min count
# Qubits	$4N^2$	$N\log(1/\epsilon)$
T-Depth	$10\log(N) + 24\log(1/\epsilon)$	$8N + 12\log(N)(\log(1/\epsilon))^2$
T-Count	$12N^2\log(1/\epsilon)$	$16N\log(1/\epsilon) + 12\log(N)(\log(1/\epsilon))^2$

Table 17.3 Explicit resource counts for block-encoding circuits of arbitrary matrices of classical data. These expressions omit subleading terms; the full expressions can be found in [296].

defined in those binary trees (the controlled Y rotations are performed *approximately* in a discrete gate set like Clifford + T, leading to a nonexact block-encoding). Tradeoffs involving the T-depth, T-count, and number of ancilla qubits are established based on the manner in which these controlled Y gates are performed. For example, the shallowest implementation [296] requires a large number of ancilla qubits, which are used to perform all the controlled Y rotations in one parallel layer, before "injecting" a subset of these ancillas into the main data qubits using a controlled SWAP network and then uncomputing the ancillas.

Dominant resource cost (gates/qubits)

The shallowest implementations are able to achieve polylog(N) depth for U_L and U_R (and hence for U_A), at the expense of $a = O(N^2)$ ancilla qubits. On the other hand, the fewest number of ancillas needed by this family of methods would be $a = n$, in which case the circuit depth would scale as $O(N^2)$. While the total circuit size must be at least $\Omega(N^2)$, the number of gates in the circuit that are T gates can be as small as $O(N)$ using the techniques in [722].

Detailed resource counts (including the constant prefactors for key metrics) and implementations of block-encodings were studied in [296]. We reproduce their resource counts optimized for minimum T-depth and for T-count in Table 17.3.

Caveats

An important caveat is that the total gate complexity of U_A must be at least $\Omega(N^2)$, reflecting the N^2 degrees of freedom in the arbitrary $N \times N$ matrix A. Thus, while A operates on a quantum system of only $\log_2(N)$ qubits, achieving depth polylog(N) requires parallel-acting gates across at least $\Omega(N^2)$ ancilla qubits. In many algorithms, it is assumed that the cost of implementing U_A is polylog(N), in order to preserve an end-to-end runtime that is polylog(N) and claims of exponential speedup. This is only defensible if the key metric is the circuit depth and if many ancilla qubits are available. Furthermore, the normal-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ization constant α can introduce poly(N) factors into an end-to-end analysis, owing to the fact that $\|A\|_F$ can be larger than $\|A\|$ by a $\sqrt{N}$ factor.

Another caveat to note is that if the matrix being block-encoded is sparse and if the values and locations of its nonzero entries can be computed efficiently, or if the matrix enjoys some structure in the data in addition to sparsity, then more efficient block-encoding methods can be employed—see Section 10.1 on block-encoding for details. In those cases, the results stated here may not be applicable.

Example use cases

In financial portfolio optimization, classical data representing average historical returns and covariance matrices for a universe of assets is needed in a quantum algorithm for optimizing a portfolio. See, for example, [328]. Similarly, in quantum machine learning based on quantum linear algebra, the algorithm often requires fast coherent access to large matrices of classically stored data.

Further reading

- An excellent overview of block-encodings and quantum linear algebra: [431].
- A detailed resource count of block-encoding with explicit circuits: [296].
- Select-SWAP QRAM and a tradeoff between qubit count and T gates: [722].
- For sparse matrices of classical data, or matrices expressed as a linear combination of Pauli matrices, more efficient methods for block-encoding exist. Asymptotic resource expressions for varying number of ancilla qubits are reported in [1084].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

18

Quantum linear system solvers

The authors are grateful to Dong An for reviewing this chapter.

Rough overview (in words)

The goal is to solve linear systems of equations with quantum subroutines. More precisely, a *quantum linear system solver* (QLSS) takes as input an $N \times N$ complex matrix A together with a complex vector b of size N, and outputs a pure quantum state $|\tilde{x}\rangle$ that is an ε-approximation of the normalized solution vector of the linear system of equations $Ax = b$. In basic versions, QLSSs do so by loading the normalized entries of the matrix A and the normalized entries of the vector b into a unitary quantum circuit, either from a quantum random access memory (QRAM) data structure, or—if the structure of A and b allows for this—by efficiently computing the corresponding entries on the fly.

Crucially, the number of algorithmic qubits of the linear system solver itself is only roughly $\log_2(N)$, which is exponentially smaller than the matrix size. While for general systems the number of QRAM qubits still scales with the matrix/vector size, QRAM encodings can be made more space efficient for sparse systems or can even be avoided when the corresponding entries are efficiently computable. The complexity of QLSSs depends on the condition number $\kappa(A) = \left\|A^{-1}\right\| \cdot \|A\|$ of the matrix A, and one then aims to give circuits with minimal quantum resource costs—such as ancilla qubits, total gate count, circuit depth, etc.—in terms of $\kappa(A)$ and the desired accuracy $\varepsilon \in (0, 1)$.

Rough overview (in math)

There are different standard input models on how the classical data from (A, b) is loaded into the quantum processing unit, which are equivalent up to small polylogarithmic overhead for general matrices. We state the complexities in

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

terms of query access of a unitary U_b preparing the $n = \lceil \log_2(N) \rceil$-qubit pure quantum state $|b\rangle = \|b\|^{-1} \cdot \sum_{i=1}^{N} b_i |i\rangle$ for $b = (b_1, \ldots, b_N)$, where $\|\cdot\|$ for vector arguments denotes the standard Euclidean norm, together with an $(\alpha, a, 0)$-block-encoding U_A of the matrix A. The QLSS problem is then stated as follows: for a triple (U_A, U_b, ε) as above, the goal is to create an n-qubit pure quantum state $|\tilde{x}\rangle$ such that

$$\left\| |\tilde{x}\rangle - |x\rangle \right\| \leq \varepsilon$$

with

$$|x\rangle = \frac{\sum_{i=1}^{N} x_i |i\rangle}{\left\| \sum_{i=1}^{N} x_i |i\rangle \right\|} \text{ defined by } Ax = b \text{ with } x = (x_1, \ldots, x_N), \tag{18.1}$$

by employing as few times as possible the unitary operators $U_A, U_b, U_A^\dagger, U_b^\dagger$, controlled versions of $U_A, U_b, U_A^\dagger, U_b^\dagger$, and additional quantum gates on potentially additional ancilla qubits. An alternative (and closely related) error metric studied in some works is based on the trace norm, requiring $\frac{1}{2} \| |x\rangle\langle x| - |\tilde{x}\rangle\langle\tilde{x}| \|_1 \leq \varepsilon$.

One way to think of the QLSS problem is that we seek the matrix inverse A^{-1}, and that this can be implemented by, for example, quantum singular value transformation (QSVT) acting on A (via its block-encoding) with a polynomial approximation of the inverse function on the interval $[\|A\|/\kappa(A), \|A\|]$. The complexity of the corresponding scheme thereby depends on the degree of the polynomial needed for a good approximation of the inverse function on the relevant interval, and as such on the condition number $\kappa(A)$, the normalization factor α, and the approximation error ε of the resulting QLSS. In fact, it turns out that the complexity of most quantum algorithms depends on the following combined quantity

$$\kappa'(A) := \kappa(A) \cdot \frac{\alpha}{\|A\|} = \alpha \cdot \|A^{-1}\|,$$

which is no smaller than $\kappa(A)$, because $\alpha \geq \|A\|$ due to the unitarity of the block-encoding. Note that in QRAM-based implementations for dense matrices A, one naturally gets $\alpha = \|A\|_F$, which then leads to linear complexity dependence on the Frobenius norm $\|A\|_F$.

As noted in [1039, 248], in general, we need not assume that A is invertible nor that it is a square matrix, but can instead use the Moore–Penrose pseudoinverse A^+ of the matrix to solve the regression problem Eq. (18.1) in a least-squares sense, in which case one needs to appropriately change the definition of $\kappa(A)$ to $\|A^+\| \cdot \|A\|$. In fact, the above QSVT-based approach directly solves this more general version of the problem [431].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost (gates/qubits)

The performance of different QLSSs is typically compared based on how their query complexity (to U_A and U_b) grows with the condition number, where a lower bound of $\Omega(\kappa(A))$ is known; see [500, 814]. Methods achieving $O(\kappa'(A))$ dependence are termed "optimal" and methods achieving $\kappa'(A)$ polylog$(\kappa'(A))$ are termed "near-optimal."[1]

The first optimal method was given in [313] (for invertible matrices), which does not directly employ the QSVT for the inverse function. Instead, it is based on discrete adiabatic methods together with quantum eigenstate filtering based on the QSVT for a minimax polynomial [689]. In particular, the adiabatic portion prepares an "ansatz" state $|x_{\text{ans}}\rangle$ for which $|\langle x_{\text{ans}}|x\rangle|^2 \geq 1/2$, using at most $O(\kappa'(A))$ (controlled) queries to U_A and U_b. Then, the eigenstate filtering step refines this state by approximately projecting it onto $|x\rangle$: one obtains the state $|\tilde{x}\rangle$ that is ε-far from $|x\rangle$ at additional query cost $O(\kappa'(A)\log(1/\varepsilon))$. The projection succeeds with probability $p \geq 1/2$, so the whole procedure must be repeated no more than twice on average. Overall, the expected number of queries made by the algorithm is Q controlled queries to each of U_A and $U_A^\dagger$ and $2Q$ queries to each of U_b and $U_b^\dagger$, where

$$Q = \kappa'(A)\Big(C + D\ln(2\varepsilon^{-1})\Big) + o(\kappa'(A)) = O\left(\kappa'(A)\log(\varepsilon^{-1})\right). \tag{18.2}$$

Here, $o(\kappa'(A))$ denotes terms growing sublinearly in $\kappa'(A)$, and C, D are constants. The algorithm operates on $n + O(1)$ qubits ($n + 5$ in the case of [313]), plus the additional qubits used for the block-encoding, discussed in more detail below. There is an additional constant quantum gate complexity for each query to U_A and U_b. For the discrete adiabatic method in [313], the constant C can be rigorously bounded as $C \leq 117{,}235$[2] and the constant D is at most 2. Note that when C is this large, the corresponding term will actually dominate the $D\kappa'(A)\log(\varepsilon^{-1})$ term for practical scenarios.

Subsequent work has given alternative methods that achieve optimal asymptotic complexity [325, 327]. Reference [325] achieves this by a small modification to and improved analysis of the adiabatic path-following method of [964]. Meanwhile, [327] replaces the step of ansatz state preparation via adiabatic methods with a simpler norm-estimation step, where one seeks a constant-

[1] Regarding the optimal ε dependence, it has additionally been claimed [313] that a complexity of $O(\kappa\log(1/\varepsilon))$ is jointly optimal in both κ and ε, based on forthcoming work by Harrow and Kothari; see also [314, Appendix A].

[2] This number is derived from applying [313, Theorem 9] with $\sqrt{2-\sqrt{2}} \times 44{,}864 \times \kappa$ steps, each of which incurs one call to the block-encoding, such that the output is guaranteed to have overlap at least $1/\sqrt{2}$ with the ideal state. Eigenstate filtering then succeeds with probability at least $1/2$; accounting for the need to repeat twice on average, one arrives at a constant 117,235, matching [572, Eq. (L2)].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

factor approximation of the Euclidean norm $\|x\|$, following that with an eigenstate filtering–like step. An optimized version of this approach was reported to have complexity following Eq. (18.2) with $C = 56$ and $D = 1.05$. Additionally, this method does not require A to be invertible, but does require b to be in the column space of A [327].

Other known QLSSs with suboptimal asymptotic complexities are based on other versions of adiabatic ansatz state preparation [964, 31, 689], QSVT [431, 744], linear combination of unitaries (LCU) [282], or variable-time amplitude amplification (VTAA) [26, 29, 248]. While the known bounds on the asymptotic complexities of these methods are slightly worse, it remains open if finite-size performance could be competitive (see, e.g., [572, 327]). Moreover, to date, the VTAA-based algorithms are the only variants that are proven to solve the generic least-squares (pseudoinverse) problem while achieving a near-optimal asymptotic scaling [248].

Note that if the matrix A is given in a classical data structure in the computational basis, then standard ways to create the block-encoding U_A make use of a QRAM structure. For general (dense) matrices A, the requirement is then size $O(N^2)$ (number of qubits) with circuit depth $O(n)$ for each query—or alternatively, as few as $O(n)$ ancilla qubits could suffice, but at the expense of using $O(N^2)$ circuit depth [496, 296]. Initializing the depth-efficient QRAM data structure will in general also take $O(N^2)$ time. However, if A is sparse, either in the computational basis [349], Pauli basis [1011], or any orthonormal basis with efficiently implementable basis transformation, there are more efficient direct constructions for block-encoding A. Moreover, for Pauli basis access, there exist randomized QLSSs with complexity scaling as the ℓ_1-norm of the Pauli coefficients [1022], completely avoiding the use of block-encodings (and as such QRAM and ancilla qubits).

Caveats

QLSSs are an important subroutine for a variety of application areas of quantum algorithms. However, it is crucial to keep track of all the quantum and classical resources required and to compare these to state-of-the-art classical methods. In particular, the following factors should be taken into account:

- The classical precomputation complexities for the eigenstate filtering routine are neglected, but can be kept efficient in practice [356].
- The rigorous upper bound on the size of the complexity constant C has been reduced by several orders of magnitude [327] since the first optimal QLSS was given in [313], but nevertheless remains larger than ideal for usage in applications where QLSS plays a heavy role. However, numerical investiga-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tions of two adiabatic methods on small random matrices gave evidence that the empirical performance of those methods is significantly better than the rigorous upper bounds [314].

- When needed, the QRAM cost can be prohibitive, if it requires the full overhead of quantum error correction and fault tolerance [496], especially for QRAMs of maximum size $O(N^2)$ qubits, required for general (dense) matrices.
- In the formulation of the QLSS problem, the pure quantum state $|x\rangle$ corresponds to the normalized solution vector of the linear system $Ax = b$. While the normalization factor $\|x\|$ can be obtained as well, this comes at the price of added query complexity scaling as $O(\kappa'(A)\varepsilon^{-1}\log(\varepsilon^{-1}))$ [327] (see also [248, Corollary 32]). This nearly achieves the lower bound of $\Omega(\kappa(A)\varepsilon^{-1})$ [327] (note that norm estimation necessarily has worse ε dependence than the QLSS itself).
- QLSSs do not produce a classical description of the solution vector x or an approximation thereof, but rather the pure quantum state $|\tilde{x}\rangle$. In order to obtain a classical approximation of the vector x, one needs to combine QLSSs with pure state quantum tomography, which can be performed using $O(N\varepsilon^{-2})$ samples. If poly(n) query-cost QRAM is also available, then the complexity can be quadratically improved in terms of the precision using optimized pure state tomography [49], or alternatively the overall complexity may be further improved using *iterative refinement* to $O(Ns^2 + Ns\kappa^2(A)/\|A\|) \cdot \text{polylog}(N/\varepsilon)$, as described in [772], where s is the maximum number of nonzero elements of A in any row or column. In the special case of Laplacian, or more generally symmetric, weakly diagonally dominant (SDD) matrices, [50] gives a quantum algorithm with complexity $\widetilde{O}(\sqrt{Ns}/\varepsilon)$ that outputs an ε-approximate solution $\tilde{x}$ with respect to the A-induced norm. (Measuring error in this norm enables their algorithm not to have a condition number dependence.) The algorithm uses QRAM and provides a subquadratic speedup compared to the classical complexity $O(N\log(1/\varepsilon))$, but uses rather different techniques compared to standard QLSS algorithms [500].
- The overall complexities $\widetilde{O}(N\kappa'(A)\varepsilon^{-1})$ and $O(Ns^2 + Ns\kappa^2(A)/\|A\|) \cdot \text{polylog}(N/\varepsilon)$ (where we generously allow poly(n) query-cost QRAM) to obtain a classical description of the solution can be compared to classical textbook Gaussian elimination–based computation, which leads to complexity $O(N^3)$ or more precisely $O(N^\omega)$ with $\omega \in [2, 2.372)$ denoting the matrix multiplication exponent. Further, QLSSs should also be compared with state-of-the-art randomized solvers. For example, the randomized Kaczmarz method [959] with standard classical access to the matrix elements returns

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

an ε-approximation of the vector x, while scaling as $O(s\kappa_F^2(A)\log(\varepsilon^{-1}))$ for s row-sparse matrices and $\kappa_F(A) = \left\|A^{-1}\right\| \cdot \|A\|_F$. Moreover, if A is s-sparse and positive semidefinite (PSD), then using the conjugate gradient method one can obtain a solution in time $\widetilde{O}(Ns\sqrt{\kappa(A)}\log(\varepsilon^{-1}))$ [481, Chapter 10.2], which can be generalized to the least-squares problem (and thus non-Hermitian matrices) at the cost of a quadratically worse condition number dependence $O(Ns\kappa\log(\kappa(A)/\varepsilon))$ by considering the modified equation $A^\dagger Ax = A^\dagger b$. As such, it seems that the QLSS may not provide a superquadratic speedup when a full classical solution is to be extracted, and even subquadratic speedups seem to be limited to a narrow parameter regime.

- Quantum-inspired methods [271, 433] that start from a classical data structure intended to mimic QRAM—allowing one to sample from probability distributions with probabilities proportional to the squared magnitudes of elements in a given row of A—give samples from an ε-approximation to the solution vector in (N-independent) complexity $O(\kappa_F^4(A)\kappa^2(A)\varepsilon^{-2})$ [924, 433], and can be used to compute an approximate solution by repeated sampling. Note that while the required data structure is classical, it might still be prohibitively expensive to build when the matrix A is huge.
- When it comes to classical methods, solvers that depend on the condition number are useful in practice whenever combined with preconditioners [888]. However, the performance of preconditioners in the quantum setting (see, e.g., [295, 925, 990, 83]) is often only heuristic, or applies only to restricted situations. This topic would benefit from further exploration.

Example use cases

- Quantum interior point methods in convex optimization and corresponding applications [610, 771].
- Quantum machine learning applications [1039, 866].
- Solving differential equations and corresponding applications, for example, for the finite element method that does not require a tomography step [777].

Further reading

- Original QLSS (termed HHL) [500].
- For an overview discussion of QLSS, see [31].
- Optimal-in-κ QLSSs are given in [313, 327, 325].
- There are also known (polynomial) speedups in case one needs a full classical description of the output vector in linear equation solving and in some regression variants [772, 266].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

19

Quantum gradient estimation

The authors are grateful to Nikitas Stamatopoulos for reviewing this chapter.

Rough overview (in words)

Estimating the gradient of a high-dimensional function is a widely useful subroutine of classical and quantum algorithms. The function's gradient at a certain point can be classically estimated by querying the value of the function at many nearby points. However, the number of evaluations will scale with the number of dimensions in the function, which can be very large. By contrast, the quantum gradient estimation algorithm evaluates the function a *constant* number of times (in superposition over many nearby points) and uses interference effects to produce the estimate of the gradient. While there are caveats related to the precise access model and the classical complexity of gradient estimation in specific applications, this procedure can potentially lead to significant quantum speedups.

Rough overview (in math)

Let $f\colon \mathbb{R}^d \to \mathbb{R}$ be a real function on d-dimensional inputs, and assume that it is differentiable at a specific input of interest, taken to be the origin $\mathbf{0} = (0, 0, \ldots, 0)$ for simplicity (the algorithm works equally well elsewhere). Let $g = (g_1, \ldots, g_d)$ denote the gradient of f at $\mathbf{0}$, that is, $g = \nabla f(\mathbf{0})$. We wish to produce a classical estimate $\tilde{g}$ of g that satisfies $|g_j - \tilde{g}_j| < \varepsilon$ for all $j = 1, \ldots, d$.

Ignoring higher-order terms, the function may be approximated near the origin as $f(x) \approx f(\mathbf{0}) + \langle g, x\rangle$, where $\langle \cdot, \cdot \rangle$ denotes the normal inner product. The original gradient estimation algorithm by Jordan [587] then considers a d-dimensional grid of points near the origin denoted by G. For simplicity, suppose on each of the d dimensions, the grid has N evenly spaced points on the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

interval $[-\ell/2, \ell/2]$, for a certain parameter ℓ related to the precision requirements of the algorithm, where N is assumed to be a power of 2. Let m be an upper bound on the magnitude of the components of g. Define $g' = Ng/2m$ to have components between $-N/2$ and $N/2$. Similarly, let $\tilde{g}' = N\tilde{g}/2m$ be the desired normalized-and-shifted output.

The quantum algorithm prepares a superposition of the grid points $x \in G$ and computes function $f(x)$ (times a constant $\pi N/m\ell$) into the phase, producing the state

$$\frac{1}{\sqrt{N^d}} \sum_{x \in G} \mathrm{e}^{\mathrm{i}\pi N f(x)/m\ell} |x\rangle \approx \frac{\mathrm{e}^{\mathrm{i}\pi N f(\mathbf{0})/m\ell}}{\sqrt{N^d}} \sum_{x \in G} \mathrm{e}^{\mathrm{i}\pi N \langle g, x\rangle/m\ell} |x\rangle \,,$$

where $|x\rangle$ denotes the product state $|l_1\rangle|l_2\rangle \cdots |l_d\rangle$, where l_j is a binary string of length $\log_2(N)$ containing a representation of the j-th component x_j of the vector x, with the identification $x_j = -\ell/2 + \ell l_j/N$. With this in mind, the latter state is rewritten as the product state, up to a global phase and normalization constant

$$\left(\mathrm{e}^{-\pi \mathrm{i} g'_1} \sum_{l_1=0}^{N-1} \mathrm{e}^{2\pi \mathrm{i} l_1 g'_1/N} |l_1\rangle\right)\left(\mathrm{e}^{-\pi \mathrm{i} g'_2} \sum_{l_2=0}^{N-1} \mathrm{e}^{2\pi \mathrm{i} l_2 g'_2/N} |l_2\rangle\right) \cdots \left(\mathrm{e}^{-\pi \mathrm{i} N g'_d} \sum_{l_d=0}^{N-1} \mathrm{e}^{2\pi \mathrm{i} l_d g'_d/N} |l_d\rangle\right).$$

Due to the approximated linearity of f, each of the product state constituents is observed to be close to a basis state in the Fourier basis (see Eq. (12.1)). By performing an inverse quantum Fourier transform (QFT) in parallel for each of the d dimensions and measuring in the computational basis, a computational basis state

$$|\tilde{g}'\rangle = |\tilde{g}'_1\rangle|\tilde{g}'_2\rangle \cdots |\tilde{g}'_d\rangle$$

is retrieved (up to an unimportant global phase), where with high probability $\tilde{g}'_j$ approximates g'_j to $\log_2(N)$ bits of precision. The coordinate $\tilde{g}_j$ is then recovered as $\tilde{g}_j = 2m\tilde{g}'_j/N$. Assuming $m = O(1)$, taking $N = O(1/\varepsilon)$ suffices to solve the problem. In a full analysis, one must make sure not to choose ℓ too large (else the linearity approximation breaks down).

In [587], the unitary U_f sending $|x\rangle \mapsto \mathrm{e}^{\mathrm{i}\pi N f(x)/m\ell}|x\rangle$ was performed using a constant number of calls to the evaluation oracle that computes an approximation to $f(x)/m$ to precision $O(\varepsilon^2/\sqrt{d})$ into an ancilla register. In [430], the precision required was improved to $O(\varepsilon/\sqrt{d})$ using finite difference formulas to put the gradient into the phase. Additionally, it was shown how U_f can be implemented using $O(\sqrt{d}/\varepsilon)$ calls to a "probability oracle" that (assuming $0 \le f(x) \le 1$) performs the map $|x\rangle|0\rangle \mapsto \sqrt{f(x)}|x\rangle|1\rangle + \sqrt{1 - f(x)}|x\rangle|0\rangle$.

The gradient estimation algorithm can be viewed as a generalization of the Bernstein–Vazirani algorithm [129], which considers binary functions f :

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$\{0,1\}^n \to \{0,1\}$, and promised that $f(x) = \langle g, x\rangle \bmod 2$ for some unknown vector g, determines g with one query to f.

Dominant resource cost (gates/qubits)

The superposition over grid points can be easily accomplished with Hadamard gates. Likewise, the inverse QFT operation is relatively cheap. The number of qubits is $O(d\log(N))$, and the number of elementary operations for each of the d parallel QFTs is polylog(N)—thus, the gate depth is independent of d, while the total gate complexity is linear in d. Additionally, an important component of the complexity comes from performing the unitary U_f, which requires implementing either an evaluation oracle or a probability oracle for the function f. If one has access to an evaluation oracle, the function must be evaluated to precision $O(\varepsilon/\sqrt{d})$. Thus, if function evaluations can be made to precision δ in circuit depth polylog$(d, 1/\delta)$, the overall circuit depth of the quantum gradient estimation algorithm will be polylog$(d, 1/\varepsilon)$, a potentially exponential speedup over the at least $\Omega(d)$ classical query complexity to learn the gradient. In the case that one has access to a probability oracle, a number of oracle calls scaling as $O(\sqrt{d}/\varepsilon)$ must be made.

For some functions, it is possible to classically compute $f(x)$ to precision δ with gate complexity poly$(d, \log(1/\delta))$. This can be turned into a quantum circuit U_f with a comparable gate complexity. For other functions, computing $f(x)$ may be much harder. For example, if $f(x)$ is defined as the output probability of a quantum circuit described by d parameters, then computing $f(x)$ to precision δ might be difficult for a classical computer, and even on a quantum computer, it generally requires $O(1/\delta)$ complexity. However, in this case, implementing a probability oracle is simple, leading to the motivation for the work of [430].

Caveats

Jordan's formulation of the algorithm [587] appears to offer a large quantum speedup by accomplishing in a single quantum query what requires $\Omega(d)$ classical queries. However, this requires a fairly strong access model where one has access to an oracle for computing the value of the function f to high precision. For an exponential speedup to be possible, precision ε must be achievable at cost polylog$(d, 1/\varepsilon)$. Unfortunately, for actual functions f that show up in applications where this is possible, it is often the case that one can classically compute the gradient much more efficiently than simply querying the value of f at many nearby points. Indeed, the "cheap gradient principle" [457, 163] asserts that (in many practical situations) computing the gradient has roughly

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the same cost as computing the function itself. This principle limits the scope of application of the large speedup of Jordan's algorithm.

By contrast, [430] shows how the gradient can alternatively be computed using a probability oracle rather than an evaluation oracle, which makes the algorithm compatible with computing gradients in the setting of variational quantum algorithms. However, $O(\sqrt{d}/\varepsilon)$ calls to the oracle are required, which represents a (much less dramatic) *quadratic* speedup compared to the strategy of using the probability oracle to estimate $f(x)$ at many nearby points and subsequently estimating the gradient classically.

Example use cases

- Convex optimization: In convex optimization, local optima are also global optima, and thus a global optimum can be found by greedy methods such as gradient descent. When one can efficiently compute the function f much more cheaply than computing its gradient, the quantum gradient estimation algorithm can give rise to a speedup over classical optimization procedures [47, 245].
- Pure state tomography: Given access to a unitary U that prepares the pure state $|\psi\rangle$, [49] utilizes the gradient estimation algorithm to estimate the amplitudes of $|\psi\rangle$ in the computational basis using an optimal number of queries to U.
- Estimating multiple expectation values: Amplitude estimation can be used to estimate an expectation value to precision ϵ at cost $O(1/\epsilon)$. In [549, 49], it is shown how the gradient estimation algorithm further allows M expectation values to be simultaneously estimated at cost $\widetilde{O}(\sqrt{M}/\epsilon)$ calls to a state preparation unitary, considered the most expensive part of the circuit.
- Computing molecular forces: While ground state energies are the object most often studied in algorithms for quantum chemistry, other interesting quantities such as molecular forces can be related to gradients of molecular energies. Reference [805] studies how the gradient estimation algorithm can be leveraged into a quantum algorithm for computing such quantities.
- Escaping saddle points: Although not the essential ingredient, the gradient estimation algorithm was used in the algorithm of [1081] for escaping saddle points.
- Variational quantum algorithms: Variational quantum algorithms involve optimizing the parameters of a quantum circuit under some cost function. The ability to estimate the gradient of the cost function with respect to the parameters might allow acceleration of this loop.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- Financial market risk analysis: In [950], the quantum gradient estimation subroutine was utilized to compute the Greeks, parameters associated with financial market sensitivity.

Further reading

See [430] for a full discussion of the state of the art with respect to the quantum gradient estimation algorithm.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

20

Variational quantum algorithms

The authors are grateful to Marco Cerezo and Xiao Yuan for reviewing this chapter.

Rough overview (in words)

The so-called noisy intermediate-scale quantum (NISQ) era is a term used to describe the regime in which the best quantum processors have fifty to a few hundred noisy qubits [843]. In this regime, one does not have enough qubits or low enough error rates to carry out fault-tolerant quantum computation, and so one is constrained to run low-depth quantum circuits. Under these constraints, structured quantum algorithms with prescribed circuits and provable guarantees are unknown. In light of this, variational quantum algorithms (VQAs) have been proposed. We remark that, despite this original setting, it would also be possible to run VQAs on fault-tolerant devices. While many VQAs have been proposed for a wide range of applications, they all share the same core primitive, which we describe below.

The main idea is to encode the target problem into an optimization task of minimizing the expectation value of some parameterized quantum circuit, or a function thereof. In each optimization step, a quantum computer is used to evaluate expectation values at chosen parameter values, which are read by a classical optimizer that updates the parameters for the next step. The motivation for this framework is to offload some of the computational complexity onto the classical optimization algorithm, with an aim for the quantum subroutines to perform classically intractable calculations.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Rough overview (in math)

Given some parameterized unitary $U(\boldsymbol{\theta})$ with adjustable parameters $\boldsymbol{\theta}$, input state ρ, measurement operator O, and function $f(\cdot)$, one evaluates $C(\boldsymbol{\theta}) = f\Big(\mathrm{Tr}\Big[OU(\boldsymbol{\theta})\rho U^{\dagger}(\boldsymbol{\theta})\Big]\Big)$ on a quantum computer, which is known as a cost function. A classical optimizer is then tasked to solve the problem $\boldsymbol{\theta}_* = \mathrm{argmin}_{\boldsymbol{\theta}} f\Big(\mathrm{Tr}[OU(\boldsymbol{\theta})\rho U^{\dagger}(\boldsymbol{\theta})]\Big)$. By careful choice of $f(\cdot)$, ρ, and O, one can encode a problem of interest such that $U(\boldsymbol{\theta}_*)$ enables an (approximate) solution to the problem. For instance, the solution could correspond to the computational basis state with which the output state $U(\boldsymbol{\theta}_*)\rho U(\boldsymbol{\theta}_*)^{\dagger}$ has the largest overlap, or to the value of $f(\mathrm{Tr}[OU(\boldsymbol{\theta}_*)\rho U(\boldsymbol{\theta}_*)^{\dagger}])$ itself. In general, one can also construct a more elaborate cost function comprising a sum of observable-dependent functions with different input states and measurement operators.

The parameterized circuit $U(\boldsymbol{\theta})$ is commonly referred to as the "ansatz circuit." The choice of cost function and ansatz are key components in designing a VQA. Namely, they should ideally satisfy the following properties:

(i) Smaller values of the cost function should correspond to better quality of solution.
(ii) The ansatz should be sufficiently expressive to contain a unitary $U(\boldsymbol{\theta}_*)$, which yields an acceptable solution.
(iii) The ansatz should lead to a trainable cost landscape in parameter space, such that a sufficiently good solution can be found efficiently by the classical optimizer.
(iv) The cost function should be classically hard to simulate, given the choice of ansatz.

It should be noted that while one would expect any VQA to satisfy the first point by design, in general, it can be hard to satisfy all of the above requirements simultaneously via theoretical guarantees or even heuristically in practice. These caveats are discussed in more detail below.

Dominant resource cost (gates/qubits)

The gate complexity is wholly dependent on the choice of ansatz. Satisfying properties ((ii)) and ((iv)) may place lower bounds on the required circuit depth. In addition, the connectivity of the device may also significantly affect the depth of the circuit. For instance, compilation of a generic two-qubit gate acting on an n-qubit state on hardware with 1D nearest-neighbor connectivity incurs $O(n)$ circuit depth.

Throughout the optimization, the cost function is evaluated at different parameter settings $\boldsymbol{\theta}$, chosen adaptively based on the outcome of prior evalua-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

tions. (In the case of gradient-based optimization, one can use the parameter shift rule [766, 918, 321, 1043] or finite difference methods.) Each evaluation of the cost function corresponds to approximating an expectation value to some additive error ε using finite measurement shots, where ε should be chosen to be sufficiently small for accurate optimization over the landscape. Specifically, it should be expected that ε is at most $O(\sqrt{\mathbb{V}_{\boldsymbol{\theta}}[C(\boldsymbol{\theta})]})$ in order to accurately distinguish arbitrarily chosen points on the parameter landscape, where $\mathbb{V}_{\boldsymbol{\theta}}$ denotes the variance over uniformly distributed parameter settings.

Caveats

The optimization of certain parameterized quantum circuits is known to be subject to the detrimental phenomena of "barren plateaus," in which deviations between different cost values with high probability (or deterministically, depending on the setting) vanish exponentially with increasing number of qubits [663, 756, 240, 534, 742, 926, 662, 396, 850]. This is often characterized by observing that $\mathbb{V}_{\boldsymbol{\theta}}[C(\boldsymbol{\theta})] = O(2^{-\beta n})$ for some $\beta > 0$ [60]. This mandates an exponential shot complexity for each evaluation of a cost value in order to reliably navigate the cost landscape (when probabilistic, this should be considered an average-case phenomenon). Note that this affects both gradient-based and gradient-free optimization strategies. Moreover, it has been found that many standard techniques to avoid vanishing gradients render the VQA classically simulable [243].

If VQAs are run on noisy devices, the effects of noise are known to severely restrict the scope for computation [12, 114, 1021, 400, 336]. This effect is amplified on devices with limited hardware connectivity, where one has to use additional circuit depth to compile generic gates [400, 1021].

Finally, in general, there is a lack of end-to-end theoretical guarantees for variational quantum algorithms. In order to show advantage over classical algorithms, at minimum one has to satisfy all of the properties laid out above. In particular, the classical parameter optimization is generally left as a heuristic subroutine. This optimization task is in general NP-hard, and can be burdened by many local minima of poor quality [154, 41]. This leads to a slow optimization process and many cost values may need to be evaluated.

Example use cases

- Quantum chemistry and condensed matter physics (ground state energy): The ground state and ground state energy of a given Hamiltonian H can be found by minimizing the cost $\langle\psi(\boldsymbol{\theta})|H|\psi(\boldsymbol{\theta})\rangle$, where $|\psi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|\psi_0\rangle$ for some input state $|\psi_0\rangle$ [833]. This is known as the variational quantum eigensolver (VQE) algorithm. A widely used ansatz for fermionic Hamiltonians

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

is the unitary coupled cluster (UCC) ansatz [982, 833, 197, 668, 786, 748, 629, 922]. Beyond direct variational optimization on a quantum computer, alternative hybrid classical-quantum algorithms have been proposed to use a variational circuit as trial states for quantum Monte Carlo [550, 539] or combining the use of VQE with density matrix embedding theory [557, 923].

- Combinatorial optimization: In the quantum approximate optimization algorithm (QAOA), combinatorial problems on bit-strings can be encoded in the Pauli-Z basis with Hamiltonian H_P [384]. By finding the state that minimizes $\langle\phi(\boldsymbol{\theta})|H_P|\phi(\boldsymbol{\theta})\rangle$, where $|\phi(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|0\rangle$, the optimal bit-string can be extracted by sampling the optimized state in the computational basis. A widely studied ansatz for this problem is the quantum alternating operator ansatz (which bears the same acronym as the algorithm), inspired by Trotterized adiabatic evolution [482]. The ansatz takes the form $U(\boldsymbol{\gamma},\boldsymbol{\beta}) = \prod_{l=1}^{p} \mathrm{e}^{-\mathrm{i}\beta_l H_M}\mathrm{e}^{-\mathrm{i}\gamma_l H_P}$ where H_M is a specific "mixing" Hamiltonian. This ansatz is known to be computationally universal (when $p \to \infty$) for certain classes of Hamiltonians [706, 780]. Moreover, under reasonable complexity-theoretic assumptions, it is known that sampling from the output of the QAOA at $p = 1$ is classically hard [380]. On the other hand, there is evidence that shallow (small p) QAOA does not perform well [194, 507, 385, 386], leading to intuition that p may need to grow with problem size to produce better approximate solutions than what can be easily found classically. Alternatively, there is some evidence that an exponential number of samples from shallow QAOA circuits may yield polynomial speedups over classical methods for finding exactly optimal solutions [179, 928]; see Section 4.2 on beyond-quadratic speedups for combinatorial optimization.
- Linear system solvers: Given matrix A and vector b encoded in a quantum state $|b\rangle$, the goal is to variationally prepare a quantum state $|x\rangle$ with amplitudes proportional to elements of the vector $x = A^{-1}b$ [187, 1060, 541]. The strategy employed is to minimize the cost $\langle\tilde{x}(\boldsymbol{\theta})|H_L|\tilde{x}(\boldsymbol{\theta})\rangle$, where $|\tilde{x}(\boldsymbol{\theta})\rangle = U(\boldsymbol{\theta})|0\rangle$ and $H_L = A^{\dagger}(I - |b\rangle\langle b|)A$. These approaches require the assumption that A has a decomposition into a sum of a small number of efficiently implementable unitaries. Here the absolute value of the cost function bounds the approximation error. A numerical study up to 30 qubits showed favorable scaling in the time to solution with respect to the matrix size, condition number, and precision [187].
- Compiling: An interesting near-term application could be to approximate a given unitary V with native gate sequence $U(\boldsymbol{\theta})$. This can lead to a compressed approximate implementation of the unitary. One option is to construct a cost function via the Hilbert–Schmidt test circuit to evaluate $1 -$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

$|\langle\Phi|V^* \otimes U(\boldsymbol{\theta})|\Phi\rangle|^2 = 1 - \left|2^{-n}\,\mathrm{tr}[V^\dagger U(\boldsymbol{\theta})]\right|^2$, where $|\Phi\rangle$ is the maximally entangled state [613].

- Quantum dynamics: VQAs can also be constructed to simulate real-time and imaginary-time evolution. We discuss two examples. First, by using the compiling primitives as outlined above, compressed gate sequences for short-time evolution can be found [119, 757, 967, 983, 159]. This in turn can also be used to fast forward and simulate evolution at long times [294, 305, 420]. Second, a line of work has considered simulating open and closed dynamics, as well as imaginary-time evolution, via McLachlan's variational principle [758], which gives a linear equation for parameter dynamics $\sum_j M_{i,j}\dot{\theta}_j = V_i$ describing the evolution of a parameterized state $|\psi(\boldsymbol{\theta})\rangle$ [684, 1074, 751]. Here, $M_{i,j}$ and V_i can both be found on a quantum computer with basic circuit primitives, leading to a hybrid classical-quantum algorithm. While a full review of variational quantum algorithms for dynamics is beyond the scope of this section, we refer the reader to more complete reviews in [1074, 241].
- Factoring: Variational methods for factoring have been proposed that exploit a mapping between the factoring problem and that of finding the ground state of an Ising Hamiltonian [39]. The authors use the QAOA ansatz and heuristically find that $p = O(n)$ rounds of the ansatz can lead to a good solution overlap for small system sizes.
- Machine learning: Here one employs a parameterized quantum circuit to construct a hypothesis family. Variational methods have been proposed for both classical and quantum data for classification [919, 766, 914, 513, 306], generative models [1008, 117, 362], autoencoders [879, 1014, 1009], and beyond [878, 547]. Specific ansatzes have been proposed in these contexts, sometimes referred to as *quantum neural networks*, in analogy with their classical counterparts. "Classically inspired" quantum neural networks have been proposed, such as perceptron-based QNNs [22, 1014, 381, 112] and a quantum analog to the convolutional neural network [306], as well as approaches based on tensor networks [451, 548].

Further reading

- See [241, 148] for extensive reviews of VQAs, including a summary of different widely studied ansatzes, applications, and challenges.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

21

Quantum tomography

The authors are grateful to Richard Kueng for reviewing this chapter.

Rough overview (in words)

In quantum tomography, often also termed quantum state estimation, we are given repeated copies of an unknown quantum state (or quantum channel) and the goal is to obtain a full classical description of the quantum state (or quantum channel) by extracting information by means of performing measurements. Here, we focus on quantum state tomography, with multiple independent and identical copies of an unknown quantum state ρ provided—that is of fixed and of known dimension—and the task is to find an estimate of the density matrix of the quantum state up to an approximation error in some distance measure (and up to some failure probability). We are then typically interested in the optimal sample complexity in terms of the number of copies n, the quantum state dimension d, the approximation error ε, and the overall failure probability δ. Additionally, algorithmic complexity aspects of the used schemes might be of importance as well.

Rough overview (in math)

Given (many copies of) an unknown quantum state ρ of known dimension d, the goal is to give a description of $\tilde{\rho}$ with the statistical estimate $\tilde{\rho} \approx_\varepsilon \rho$, up to some distance measure with corresponding approximation parameter $\varepsilon \geq 0$. This is achieved by extracting classical information by applying measurements $\mathcal{M}^n(\cdot)$ via $\rho^{\otimes n}$. To start with, one has to distinguish tomography schemes based on different types of measurements used. This includes in particular:

(i) Independent and identical (IID) measurements, where the choice of measurement $\mathcal{M}^n = \mathcal{M}^{\otimes n}$ is fixed and the same for each copy.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

(ii) Adaptive measurements, where the choice of measurement $\mathcal{M}_2$ on the second copy can depend on the outcomes of measurement $\mathcal{M}_1$ on the first copy, and so on.

(iii) Entangled measurements, where one measurement $\mathcal{M}^k$ with $1 < k \leq n$ is performed on k copies at once.

Further, if one has some information about the type of quantum state provided, then tomography schemes can become more efficient. This includes, for example, pure state tomography, low-rank-k state tomography, matrix product state tomography, or ground/thermal state tomography of Hamiltonians (some references on tight schemes are given later on). For some schemes, one a priori has certain information about the state in question and under this assumption the scheme is then promised to work (e.g., low-rank tomography [462]). Other schemes work generally, but are only *a posteriori* guaranteed to be more efficient if the unknown state happens to be approximately of the type sought after (e.g., matrix product state tomography [317]). Finally, for maximum likelihood estimates or Bayesian statistical estimates and alike, priors could be added as well.

Note that the best understood case of pure state tomography can also be used for general quantum states, if one has access to the relevant purification. Specifically for pure state tomography, one then also needs to specify in what form access is given to the quantum state. Possible access models for pure state tomography include:

- Via samples of computational basis measurements $p(x) = \langle x|\rho|x\rangle$ for estimating the probabilities in the computational basis (not yet the pure state amplitudes).
- Via copies of the state that can be processed before measurement.
- Via the state preparation unitary $U|0^n\rangle\langle 0^n|U^\dagger = \rho$ (with ρ pure).
- Via the controlled version of aforementioned state preparation unitary U.
- Via aforementioned state preparation unitary U and its inverse $U^\dagger$.

Typically studied distance metrics to measure closeness of the statistical estimate to the true quantum state are the trace distance $T(\rho,\sigma) = \frac{1}{2}\mathrm{tr}[|\rho-\sigma|]$, the quantum fidelity $F(\rho,\sigma) = \left(\mathrm{tr}\left[\left|\sqrt{\rho}\sqrt{\sigma}\right|\right]\right)^2$, and for pure quantum states also the ℓ_2-norm of the difference $\||\psi\rangle - |\phi\rangle\|$ of the pure states $|\psi\rangle, |\phi\rangle$ corresponding (up to global phase) to ρ and σ, respectively.

Dominant resource cost (gates/qubits)

Besides some potential ancilla qubits (few for typical tomographic schemes), the number of qubits is fixed by the dimension of the quantum state (of course,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

whenever entangled measurements are used, the corresponding number of copies is needed). As such, the sample complexity is typically the relevant figure of merit. In the following, the notation $\Theta(\cdot)$ stands for simultaneous upper $O(\cdot)$ and lower $\Omega(\cdot)$ bounds on the asymptotic sample complexity, and the variant $\widetilde{\Theta}(\cdot)$ denotes the same up to factors that scale polylogarithmically in the relevant parameters. Tight sample and query complexity characterizations, in terms of an approximation error $\varepsilon \in [0, 1]$, then include the following noteworthy results:

- $\widetilde{\Theta}(d\varepsilon^{-2})$ sample complexity for pure state tomography in ℓ_2-norm up to global phase [49] ([610] gave an algorithm with similar complexity, but requiring a state preparation unitary). The main idea is to use computational basis measurements to recover the absolute values of the amplitudes and then create some interference pattern for learning the phases.
- $\widetilde{\Theta}(d\varepsilon^{-1})$ query complexity for pure state tomography in ℓ_2-norm with access to controlled state preparation unitary and its inverse [49], featuring a quadratic speedup in $1/\varepsilon$ reminiscent of amplitude estimation. The achievability results are based on the subroutine of quantum gradient estimation via an unbiased version of quantum phase estimation. Note that [267] used an alternative simpler algorithm based on iterative refinement and amplitude amplification that achieves the same query complexity but comes with improved gate complexity.
- $\Theta(dk^2\varepsilon^{-2})$ sample complexity for rank-k state tomography in trace distance with IID measurements [656, 478, 471]. The achievability results are based on low-rank matrix recovery techniques, where semidefinite programs have to be solved for reconstructing the quantum state from the collected measurement statistics. Note that the special case $k = 1$ corresponds to pure state tomography as in the setting of the first bullet point.
- $\widetilde{\Theta}(dk\varepsilon^{-2})$ sample complexity for rank-k state tomography in trace distance with entangled measurements [807, 478, 1075]. The achievability results are based on representation-theoretic techniques around the Schur transform.
- $\widetilde{\Theta}(dk\varepsilon^{-1})$ query complexity for rank-k state tomography in trace distance with access to controlled state preparation unitary of a purification and its inverse [49], featuring a quadratic speedup in $1/\varepsilon$ reminiscent of amplitude estimation.

For variations of the above, additional results in terms of lower and upper bounds are known. The derivations of the sample complexity lower bounds are often based on information-theoretic methods, exploiting the monotonicity of quantum-entropy-based measures. For sample complexity upper bounds, it is in practice additionally important that the algorithmic complexities of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the underlying schemes become efficient (in particular for entangled measurements performed on all n copies at once). Relevant metrics for the algorithmic complexity include quantum gate depth and number of measurement outcomes needed, as well as runtime and memory requirements of the classical postprocessing stage. We refer to [723] for a recent discussion on these computational aspects.

Caveats

As shown by the presented information-theoretic lower bounds, the sample complexity for general quantum state tomography grows exponentially in the number of qubits. As such, whenever quantum tomography is invoked as a subroutine in quantum algorithms, one has to carefully analyze if this step does not eliminate any claimed speedups of the quantum algorithm compared to state-of-the-art classical methods. One also has the inverse polynomial scaling in terms of the approximation parameter from the finite statistics, which is often prohibitively expensive for certain applications.

Additionally, on top of sample complexity for tomography schemes, the accompanying gate complexity should be considered as well. We refer to [49] for a discussion.

An alternative is to resort to only revealing partial classical information about quantum states, which might still be informative for the (algorithmic) task at hand. One such example with favorable scaling is shadow tomography, where the task is to not estimate the density matrix itself, but (very) many observables thereof. Shadow tomography—also known as quantum data analysis—can achieve exponential sample complexity improvements in terms of Hilbert space dimension compared to full state tomography and is guaranteed to yield exponential improvements in the number of target observables (compared to directly measuring all of them sequentially). The strongest result of this kind [2, 5, 209] requires entangled measurements across many state copies, as well as prohibitively large gate counts. More hardware-friendly protocols have been derived, known as classical shadow tomography [540, 823, 373]. In more detail, there exist algorithmically efficient and universal schemes that can simultaneously ε-approximate M linear functions $\mathrm{tr}[O_i\rho]$ of an unknown quantum state ρ by only using $O(\log(M)\cdot \max_i \|O_i\|_s^2\varepsilon^{-2})$ IID measurements. Note the scaling with $\log(M)$ instead of the standard M scaling. The shadow norm term $\|O_i\|_s^2$ scales in general as d, leading to the worst-case query complexity $O(d\log(M)\varepsilon^{-2})$. However, for observables with bounded Hilbert–Schmidt norm or for local observables, the overall dimension-free query complexity $O(\log(M)\varepsilon^{-2})$ is achievable.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Example use cases

Quantum tomographic or related data collection schemes are omnipresent in quantum algorithms. Some applications include:

- Quantum linear system solvers that output the full classical solution vector, where such solvers are, for example, employed for quantum interior point methods or for solving differential equations.
- Classical data about quantum states for variational quantum algorithms.
- Characterizing the performance of physical devices.
- Probing entanglement dynamics throughout a quantum simulation.
- Characterizing quantum processes.

Further reading

- Short perspective article, entitled "Focus on quantum tomography" [87].
- Recent overview on query complexity aspects [49].
- Recent overview on computational complexity aspects [723].
- Shadow tomography of quantum states [2].
- Review article on classical shadows and randomized measurements [373].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

22

Quantum interior point methods

The authors are grateful to Sander Gribling for reviewing this chapter.

Rough overview (in words)

Interior point methods (IPMs) are a type of efficient classical algorithm for solving convex optimization problems such as linear programs (LPs), second-order cone programs (SOCPs), and semidefinite programs (SDPs). IPMs are the basis for effective optimization software tools (e.g., [355, 38]), which are widely used for solving convex optimization problems that arise in industry. They are called *interior* point methods because, in contrast to the simplex method, they iteratively generate a sequence of points that lie in the interior of the convex region; this sequence of points is guaranteed to rapidly approach the optimal point (which, when it exists and the objective function is convex, is guaranteed to lie at the boundary of the convex region). At each iteration, the next point is produced by solving a system of linear equations. See, for example, [1053, 1052, 797, 438] for context on how IPMs fit into the history of methods for optimization.

Quantum interior point methods (QIPMs) are quantum algorithms that leverage a similar approach as classical IPMs, but perform certain aspects of the algorithm in a quantum manner. For example, QIPMs were first introduced in [610], where the quantum algorithm is identical to classical IPMs, except that it determines the next point using a quantum linear system solver (QLSS) combined with quantum state tomography, rather than a classical linear system solver. Subsequent work has explored other forms of quantizing classical IPMs that do not rely on the QLSS [51, 69].

Classical IPMs are generally efficient in the sense that they can solve convex optimization problems in time scaling as a polynomial in the number of vari-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ables. The exact degree of the polynomial depends on which kind of convex optimization problem is being solved, as well as certain choices about the IPM. Since QIPMs often rely on state tomography, they are generally expected to require time that scales at least linearly in the number of variables, and lower bounds along these lines are known [48]; thus, *the best one can hope for is a polynomial speedup* over classical IPMs. The exact runtime of the quantum algorithm depends on instance-specific parameters, such as the condition number of matrices that appear during the course of the algorithm, which makes it difficult to determine whether a speedup exists in practice.

Rough overview (in math)

For simplicity, we focus on LPs, the simplest kind of optimization problem where QIPMs can be applied. An LP is specified by an $m \times n$ matrix A, an n-dimensional vector c, and an m-dimensional vector b, and it is given by

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \langle c, x \rangle \\ &\text{subject to } Ax = b \\ &\qquad x_i \geq 0 \text{ for } i = 1, \ldots, n \end{aligned} \quad , \tag{22.1}$$

where $\langle u, v \rangle$ denotes the standard dot product between vectors u and v.

The function $\langle c, x \rangle$ is called the objective function, and a point x is called feasible if it satisfies $Ax = b$ and $x_i \geq 0$ for all i. Inequality constraints of the form $Ax \leq b$ can be handled by introducing slack variables. We denote the feasible point that optimizes the objective function by x^*.

An important concept in mathematical optimization is duality, where given one optimization problem, an equivalent "dual" optimization problem can be generated through the method of Lagrange multipliers (see [180, Section 5]). The dual of the LP in Eq. (22.1) is given by

$$\begin{aligned} &\max_{y \in \mathbb{R}^m} \langle b, y \rangle \\ &\text{subject to } A^\intercal y + s = c \\ &\qquad s_i \geq 0 \text{ for } i = 1, \ldots, n \end{aligned} \quad . \tag{22.2}$$

Alternatively, one can drop the s variable and constraints that s_i are positive, and simply write $A^\intercal y \leq c$. Denote the optimal feasible points for the dual by (y^*, s^*).

It can be shown that the optimal point lies at the boundary of the feasible region and satisfies the relationship $x_i s_i = 0$ for all i. A key concept in IPMs is the *central path*, a set of points parameterized by $\mu > 0$. The central point with parameter μ is the feasible point for which $x_i s_i = \mu$ for all i. In general,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

this point will be in the interior of the feasible region, but as $\mu \to 0$, the central path approaches the optimal point on the boundary.

The most effective classical IPMs are "primal-dual path-following methods," which generate a length-T sequence of primal-dual point pairs $(x^{(t)}, y^{(t)}, s^{(t)}) \in \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^n$ for $t = 0, \ldots, T-1$ that approximately follows the central path toward the optimum. Given $(x^{(t)}, y^{(t)}, s^{(t)})$, the point $(x^{(t+1)}, y^{(t+1)}, s^{(t+1)}) = (x^{(t)} + \Delta x, y^{(t)} + \Delta y, s^{(t)} + \Delta s)$ is formed by solving the following linear system of equations, which is called the *Newton system*, as it corresponds to one iteration of Newton's method.

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^\intercal & I \\ S & 0 & X \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta s \end{pmatrix} = \begin{pmatrix} b - Ax^{(t)} \\ c - A^\intercal y^{(t)} - s^{(t)} \\ \sigma \frac{x^{(t)\intercal} s^{(t)}}{n} \mathbf{1} - X s^{(t)} \end{pmatrix}, \tag{22.3}$$

where $\sigma < 1$, $\mathbf{1}$ denotes the all 1s vector, and $S = \mathrm{diag}(s^{(t)}), X = \mathrm{diag}(x^{(t)})$ are diagonal $n \times n$ matrices formed from the entries of $s^{(t)}$ and $x^{(t)}$. Note that there are alternative ways to formulate the Newton system (see, e.g., [70, 68]). To understand Eq. (22.3), note that if the point $(x^{(t)}, y^{(t)}, s^{(t)})$ is feasible, then the first two entries on the right-hand side are zero. Furthermore, if it is on the central path, then $Xs^{(t)} = \frac{x^{(t)\intercal} s^{(t)}}{n} \mathbf{1}$, so if we were to choose $\sigma = 1$, then the entire right-hand side would be zero, and the solution to the system would be $\Delta x = \Delta y = \Delta s = 0$. If instead we set $\sigma = 1 - \delta$ for sufficiently small δ, the solution will correspond to taking a small step along the central path in the direction of decreasing μ. Technically, we do not exactly follow the central path, but it can be guaranteed that the sequence of points stays within a small neighborhood of it. As $\mu \to 0$, the central path approaches the optimal point (x^*, y^*, s^*), so by following the path toward $\mu = 0$, a classical or quantum IPM can guarantee success.

The classical IPM can solve the Newton system exactly using Gaussian elimination in $O(n^3)$ operations, or it can solve the system approximately using a variety of iterative solvers such as the conjugate gradient method. In contrast, the standard approach for a QIPM is to solve the Newton system by using a QLSS to repeatedly prepare the $O(\log(n))$-qubit state $|\Delta x, \Delta y, \Delta s\rangle$ whose amplitudes encode the solution to the Newton system. By preparing many copies, the algorithm can perform (pure state) quantum state tomography to yield an estimate $(\overline{\Delta x}, \overline{\Delta y}, \overline{\Delta s})$ for the amplitudes $(\Delta x, \Delta y, \Delta s)$ to some desired precision ξ (in 2-norm), that is,

$$\|(\overline{\Delta x}, \overline{\Delta y}, \overline{\Delta s}) - (\Delta x, \Delta y, \Delta s)\| \leq \xi \|(\Delta x, \Delta y, \Delta s)\| .$$

Due to the tomography step, the QIPM is only able to generate solutions to the Newton system that are *inexact*. There has been some question in the liter-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ature whether the (classical or quantum) IPMs with the fastest guaranteed convergence rate (i.e., the number of iterations needed to reduce μ to ϵ) are applicable even when inexact solutions are used, as this causes intermediate points to be (slightly) infeasible [70]. However, if ξ is sufficiently small, the method appears to work empirically even using the inexact solutions that would be output by a quantum solver [328]. Alternatively, there exist workarounds [70] that ensure feasibility is maintained even when linear systems are solved inexactly, at the expense of some additional classical cost.

The IPMs and QIPMs for SOCPs [612, 68] are quite similar to those for LPs described above: the main difference is that the matrices X and S are no longer strictly diagonal matrices. QIPMs have also been proposed for SDPs [610, 70, 537], which are more complex but have more expressive power; here, additional considerations must be taken to guarantee that the intermediate solutions continue to be symmetric even after experiencing errors due to tomography.

The above exposition represents the original approach to quantizing the classical IPM, which has so far garnered the most study. An alternative to this approach was proposed in [51], which focuses on the case that the LP constraint matrix A is "tall," that is, $m \gg n$. As above, they follow the central path to the optimal point; however, they adopt a primal-only approach, where the Newton linear system takes on the form $(B^\intercal B)g = h$, with g and h length-n vectors and B an $m \times n$ matrix. Rather than using the QLSS and quantum state tomography, their quantum algorithm performs a Grover search–like step to identify the "important" rows of B and thus produce an $O(n) \times n$ matrix $\tilde{B}$ for which $\tilde{B}^\intercal \tilde{B} \approx B^\intercal B$. This enables a quadratic speedup in the large parameter m. To obtain the right-hand side vector h, which is the gradient of the objective function, they require the multivariate mean-estimation algorithm of [310], which is related to the quantum gradient estimation primitive developed in [587, 430]—this is key for avoiding a dependence on the condition number of $B^\intercal B$. Matrix inversion is then performed classically at cost polynomial in n, independent of m and not depending on the condition number of any matrix.

Meanwhile, another quantum algorithm inspired by IPMs was proposed in [69]. Where the standard QIPM encodes the variable x into the amplitudes of the quantum state, requiring readout with quantum state tomography, the method of [70] encodes the components of x into separate binary registers, truncated to some finite number of bits of precision. It constructs a Hamiltonian, parameterized by μ, whose ground state is a wavefunction localized near the associated point on the central path. By slowly decreasing μ and invoking the adiabatic theorem, the wavepacket follows the central path to $\mu = 0$, where the optimal point can be recovered by a measurement. Thus, the main primitive required is time-dependent Hamiltonian simulation.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Dominant resource cost (gates/qubits)

The outer loop of QLSS-based QIPMs is purely classical; at each iteration a small step is taken to form the next point in the sequence. For LPs, SOCPs, and SDPs, the number of iterations T required to yield a point for which the objective function is within ϵ of optimal is $O(\sqrt{n}\log(1/\epsilon))$. The main cost of each iteration is solving the Newton system. In the complexity statements that follow, we assume the number of constraints m is on the order of the number of degrees of freedom (i.e., $m = O(n)$ in the case of LPs and SOCPs, and $m = O(n^2)$ in the case of SDPs).

The QIPM solves the Newton system by preparing many copies of the state corresponding to the solution to the linear system. This state can be prepared in time $\text{polylog}(n)\cdot\zeta\kappa$, where κ is the condition number of the matrix in Eq. (22.3) and ζ is the ratio $\|\cdot\|_F/\|\cdot\|$ of the Frobenius and spectral norms of the matrix, assuming that one can perform a block-encoding of the Newton matrix in $\text{polylog}(n)$ time, a task that requires access to large-scale quantum random access memory (QRAM).[1] For LPs and SOCPs, the number of copies that must be prepared scales as $O(n/\xi^2)$ when using the basic version (see [610, Section 4] and [328, Section IVD]) of pure state tomography that simply measures each copy in the computational basis. A more recent and complex version of tomography [49] can achieve this task using $O(n/\xi)$ copies along with additional gates. For SDPs, since the variables are matrices rather than vectors, the number of copies is $O(n^2/\xi^2)$ or $O(n^2/\xi)$. Overall, using the more efficient version of tomography and ignoring the additional gates, the runtime of the QIPM is expected to scale as

$$\begin{aligned} \text{LP, SOCP:} \quad & \widetilde{O}\left(\frac{n^{1.5}\zeta\kappa}{\xi}\log(1/\epsilon)\right) \\ \text{SDP:} \quad & \widetilde{O}\left(\frac{n^{2.5}\zeta\kappa}{\xi}\log(1/\epsilon)\right), \end{aligned} \tag{22.4}$$

where κ denotes the maximum condition number, ζ the maximum ratio of Frobenius to spectral norm, and ξ the minimum tomographic precision required across all iterations. There may be an additional purely classical cost of $O(n^{2.5})$ for LPs/SOCPs and $O(n^{4.5})$ for SDPs, deriving from classical matrix-vector multiplications necessary for setting up the Newton system at each iteration.

[1] It is worth emphasizing that the origin of the dependence on the Frobenius norm of the Newton matrix here is the normalization factor that arises when block-encoding a dense classical matrix. If the matrix were sparse or had some compact representation, this normalization factor could potentially be improved—but for Newton matrices in QIPMs we do not expect this to be the case.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

In the worst case, it may be necessary to take ξ as small as $O(1/\kappa)$, and ζ can be as large as $\sqrt{n}$ (SOCP/LP) or n (SDP)—complexity statements in the literature, such as [70], often assume these worst-case values for those parameters, but we refrain from doing so as these worst-case values may be overly pessimistic in practice. The hidden constant prefactors are dependent primarily on the implementation of the QLSS and tomography. It is clear that the viability of the QIPM is highly dependent on the value and scaling of the parameters κ and ξ. Unfortunately, it is believed that for some LP/SOCP/SDP instances, the value of κ will diverge as the target precision ϵ is made smaller, perhaps as $O(1/\epsilon)$ [612, 70], although this may not be the case in every instance (see, e.g., the numerical results of [328]).

The QIPM only requires a register of $O(\log(n))$ qubits to hold the solution of the linear system; however, achieving the runtimes quoted requires queries to QRAM. In this case, the explicit QRAM circuits that achieve shallow depths of $O(\log(n))$ necessarily require $O(n^2)$ total gates across $O(n^2)$ total qubits.

The alternative approach of [51] is best suited for the case where $m \gg n$, and requires $\sqrt{m} \cdot \text{poly}(n, \log(1/\epsilon))$ queries to the entries of the matrix A, where the $\sqrt{m}$-dependence fundamentally comes from Grover-like primitives with quadratic speedup. Like the standard QIPM formulation, this approach requires a QRAM to implement the queries in polylog(m) time. However, since it does not use QLSS or tomography, it avoids polynomial dependence on the instance-specific parameters κ and $1/\xi$.

Caveats

There are several important caveats that must be considered when evaluating a speedup claimed by QIPM.

- Even in a best case scenario, the quantum speedup is at most polynomial (and even subquadratic). Since quantum computation requires significant constant-factor overheads due to slower clock speeds and error correction, the value of n for which a QIPM would be faster than a classical IPM on actual hardware is likely to be large (see [328] for further discussion).
- Since n must be large for a quantum speedup to be obtained, a very large QRAM, corresponding to millions or billions of (logical) qubits, would be needed for any speedup to be realized.
- QIPMs are most effective when the matrices that need to be inverted over the course of the algorithm are well conditioned, due to their reliance on the QLSS. However, when the condition number κ is small, iterative classical methods may also be effective, limiting the advantage of the quantum algorithm. In particular, a linear system with $O(n)$ dense constraints on n

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

variables can be solved to error ξ in time $O(n\zeta^2\kappa^2 \log(1/\xi))$ using the randomized Kaczmarz method [959]. In comparison, the QIPM utilizes QLSS and tomography to solve the same task (once per iteration) in time $O(n\zeta\kappa/\xi)$. Even if $\xi = \Omega(1)$, this limits the magnitude of the quantum speedup to a factor of $O(\zeta\kappa)$. Thus, for the quantum speedup to be maximized, κ can be neither too small nor too large. While we are not aware of any IPM implementations based on the Kaczmarz method, its complexity allows for clean comparison with quantum algorithms involving the QLSS for dense matrices, since both depend directly on the quantity $\zeta\kappa$. Here it is also worth mentioning that there exist other approximate classical linear system solvers for which the complexity depends on κ, but not on ζ. One example is the conjugate gradient method [713]. Another straightforward example is to solve the system $Gu = v$ by finding a degree-$O(\kappa \log(1/\xi))$ polynomial approximation $p(x) \approx 1/x$, and then computing $p(G)v \approx G^{-1}v = u$ via a sequence of $O(\kappa \log(1/\xi))$ matrix-vector products—this is a classical analog of the quantum approach based on the quantum singular value transformation [431]. Classically, each matrix-vector product costs $O(n^2)$ when G is dense and $O(ns)$ for when G is s-sparse.

- If the matrices that define the convex problem have a certain structure (e.g., sparsity), this could be exploited to potentially reduce the overhead from block-encoding—in particular, the value of ζ and the size of the QRAM required. However, this can help the quantum algorithm only to a limited extent, as the vectors $(\Delta x, \Delta y, \Delta s)$ will still be dense and reading out estimates for all $O(n)$ amplitudes with quantum tomography will be necessary.

Example use cases

- Portfolio optimization, the canonical optimization problem that appears in finance, can be formulated as an SOCP and solved with a QIPM; a study of the condition number of the matrices that appear in this application was consistent with a small quantum speedup [611]; however, a follow-up study did not replicate this finding [328] and also pointed out that in any case large constant-factor overheads would make achieving practical advantage challenging.
- Support vector machines, a common task in machine learning, can be reduced to SOCPs and solved with a QIPM; a study of the condition number of the matrices that appear in this application was consistent with a small quantum speedup [612].
- Sample-efficient protocols for mixed-state tomography reduce the problem of reconstructing an estimate of the quantum state to solving an SDP. This

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

SDP could be solved with a QIPM (note that the tomography needed within the QIPM is always on *pure states* and does not require solving an SDP, thus avoiding an issue of circular logic).

- Nonconvex optimization is often solved approximately by relaxing the problem into a convex problem like an SDP. For example, the MAX-CUT problem is a combinatorial optimization problem over the nonconvex space $\{+1, -1\}^n$, but by solving the associated SDP relaxation and rounding, an approximate solution can be obtained.

Further reading

- See Boyd and Vandenberghe [180] for an accessible book on convex optimization including (classical) IPMs.
- QIPMs are an active area of research. A QIPM for LPs and SDPs was originally proposed by Kerenidis and Prakash in [610]. This was followed up by a QIPM for SOCPs in [612], along with numerical simulations for specific applications [612, 611]. Later, [70] pointed out a potential error in the convergence analysis of previous works, and they presented two possible workarounds called the "inexact-infeasible" and "inexact-feasible" IPMs. Note also the work in [537] for another way to avoid this issue, giving a QIPM for SDP.
- See [51, 69] for quantum methods related to IPMs that do not rely on the QLSS.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

23

Multiplicative weights update method

The authors are grateful to Sander Gribling for reviewing this chapter.

Rough overview (in words)

The multiplicative weights update (MWU) method is an algorithmic strategy, sometimes referred to as a "meta-algorithm," with varying applications in classical and quantum algorithms. Reference [58] gives an overview of the MWU strategy. The introductory example problem where the MWU method is used is the problem of making predictions for a binary outcome given advice from a panel of n "experts." The MWU approach assigns a weight to each of the n experts, and the weight is reduced by a multiplicative factor whenever the expert makes an incorrect prediction. The outcome of the process can be shown to give an approximately optimal strategy.

This general approach can be applied to convex programs including linear programs (LPs) and semidefinite programs (SDPs). The SDP version generalizes the MWU method to allow for matrix-valued weights and matrix-valued costs. These weight matrices are positive semidefinite operators with trace equal to one, that is, density matrices. In fact, the states that arise in the SDP-solving algorithm are Gibbs states. Thus, they can be naturally represented as quantum states on a logarithmic number of qubits and generated through the process of Gibbs sampling. The existence of fast Gibbs samplers can lead to a quantum speedup in certain circumstances.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Rough overview (in math)

We present an example problem. Let $\mathbf{1}$ denote the all-ones vector. Consider the following set of linear constraints on the vector $x = (x_1, \ldots, x_n) \in \mathbb{R}^n$

$$\begin{aligned} \langle a^{(j)}, x\rangle &\geq 0 \qquad j = 1, \ldots, m \\ \langle \mathbf{1}, x\rangle &= 1 \\ x_i &\geq 0 \qquad i = 1, \ldots, n \end{aligned}$$

for m fixed vectors $a^{(j)} \in \mathbb{R}^n$ with entries in $[-1, 1]$, for $j = 1, \ldots, m$, where $\langle \cdot, \cdot \rangle$ denotes the standard dot product between vectors. Suppose we are given a value of ϵ and promised either that there is no choice of x that satisfies all the constraints or that there exists an x^* such that $\langle a^{(j)}, x^*\rangle \geq \epsilon$ for all j, with $\langle \mathbf{1}, x^*\rangle = 1$ and $x_i^* \geq 0$ for all i. We wish to determine which is the case and find a vector x^* in the second case. This problem is equivalent to the machine learning problem of finding a linear classifier for a set of m labeled n-dimensional training points, similar to a support vector machine [58, 697]. The problem is also similar to the form of an LP and to the problem of solving for the optimal point of a zero-sum game [58, 46], and the MWU meta-algorithm can also be straightforwardly applied to solve these problems.

A classical solution to this problem is given by the multiplicative weights method [58]. The algorithm iteratively updates the vector x, with initialization $x = \mathbf{1}/n$. At each iteration, the algorithm finds a constraint j for which $\langle a^{(j)}, x\rangle < 0$ (or if no such j exists, it terminates and outputs x). Let $\eta = O(\epsilon)$ be a fixed constant. Once j is found, the entries of the vector x are updated according to

$$x_i \leftarrow \frac{x_i e^{\eta a_{ij}}}{\sum_\ell x_\ell e^{\eta a_{\ell j}}}, \tag{23.1}$$

where a_{ij} denotes entry i of vector $a^{(j)}$, and the denominator works to enforce $\langle \mathbf{1}, x\rangle = 1$. By upweighting x in the direction of the violated constraint $a^{(j)}$, this update rule brings the x closer to satisfying the constraint. The magic of the multiplicative weights method is that the promise problem described above can be solved after only $O(\log(n)/\epsilon^2)$ iterations [58]. By searching for a violated constraint using a Grover search, the runtime of each iteration can be sped up quantumly, giving rise to polynomial speedups for solving zero-sum games and LPs more generally [46].

In the analogy to a panel of experts, we may view the above problem as follows. Each expert $i \in [n]$ produces a prediction for each of $j \in [m]$ data points, denoted by a_{ij}. We wish to produce a weighting x over the n experts such that the weighted majority of the n experts yields a positive value for all m data points. The multiplicative weights method solves this iteratively by be-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ginning with a uniform weighting over the n experts, and repeatedly observing an index j where the weighted majority misclassifies (i.e., predicts a negative value for) data point j, assessing a multiplicative penalty of $e^{\eta a_{ij}}$ to the weight of expert i. In a machine learning context, the weighting of experts produced by the algorithm can then be used as a classifier that allows us to predict a label for a new data point, by following the opinion of the weighted majority of the experts.

The *matrix* MWU method generalizes the n-dimensional vector x to an $n \times n$ symmetric matrix X. An example problem generalizing the above is

$$\begin{aligned} \langle A^{(j)}, X \rangle &\geq 0 \qquad j = 1, \ldots, m \\ \langle \mathbf{I}, X \rangle &= 1 \\ X &\geq 0 \,, \end{aligned}$$

where $A^{(j)}$ are fixed symmetric constraint matrices and the notation $\langle U, V \rangle := \mathrm{Tr}(UV)$ generalizes the dot product from vectors to matrices. Here $\mathbf{I}$ denotes the identity matrix, and $X \geq 0$ denotes that X is positive semidefinite. The problem above is related to the general form of an SDP, and the matrix MWU approach can be applied to solve SDPs. Note that we recover the vector example if we specify that the matrices $A^{(j)}$ and X are diagonal. The final two constraints indicate that X is a density matrix and is associated with a quantum state on $\log_2(n)$ qubits. When X is updated by a generalization of the rule in Eq. (23.1), then at every iteration of the MWU method, X will be a Gibbs state for a certain Hamiltonian that is a weighted sum of the symmetric constraint matrices $A^{(j)}$. Thus, the quantum state X can be prepared on a quantum computer using algorithms for Gibbs sampling. Taking this approach, quantum algorithms can achieve guaranteed polynomial speedups for performing an iteration of the MWU method compared to classical approaches, and it is conceivable that larger speedups could be available if the associated quantum systems admit faster-than-worst-case Gibbs sampling.

Dominant resource cost (gates/qubits)

The MWU method, both in the classical and quantum setting, consists of some number T of iterations, where each iteration updates a classical data structure. In typical applications, $T = \mathrm{poly}(\log(n)/\epsilon)$, where n is the problem size and ϵ is a precision parameter related to how close to optimal the solution has to be. This contrasts with other approaches to solving optimization problems, such as interior point methods, for which the number of iterations can scale as $O(\mathrm{poly}(n)\log(1/\epsilon))$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Each iteration typically takes $\text{poly}(n, m, 1/\epsilon)$ time and is carried out with subroutines that can often be sped up with quantum algorithms. These subroutines can include Grover search / amplitude amplification and, in the case of the matrix MWU method, Gibbs sampling, which end up dominating the quantum cost of the algorithm.

Here it is important to point out that, especially in the quantum case, the MWU method can benefit from keeping an implicit representation of the n-dimensional vector x (or in the case of matrix MWU, the $n \times n$ matrix X). For instance, in the example problem above, we need not explicitly write down the vector x; rather, we can keep track of the indices $j_1, j_2, \ldots, j_t$ corresponding to the penalties assessed at iterations $1, 2, \ldots, t$. These indices can be organized into a t-sparse vector $y \in \mathbb{R}^m$ from which x is defined implicitly by $x_i \propto e^{\eta \sum_{j=1}^m a_{ij} y_j}$. In the context of optimization problems like LPs and SDPs, the vector y can often be related to the *dual* version of the optimization problem (see, e.g., [45]), where the goal is to find an optimal y, and each value y_j may be interpreted as the weight assigned to decision $j \in [m]$. Given y, the Gibbs sampling primitive can then produce a quantum state on $O(\log(n))$ qubits encoding the vector x. At iteration $t + 1$, this quantum state is used to find an index j_{t+1} corresponding to a violated constraint without ever explicitly writing down the vector x. This implicit representation is essential if the quantum algorithm is to achieve complexity sublinear in n. The same situation arises in algorithms for SDP based on matrix MWU, where Gibbs sampling is used to produce a $O(\log(n))$-qubit mixed quantum state $\rho = e^{-H}/\text{tr}(e^{-H})$, where on iteration $t+1$, the Hamiltonian $H = \sum_j y_j A^{(j)}$ is given by a weighted sum of at most t distinct input matrices. By keeping track only of the sparse vector $y \in [m]$, one avoids needing to write down the n^2 entries of ρ. In fact, ignoring dependence on ϵ, the Gibbs state ρ can typically be prepared using only $\widetilde{O}(s\sqrt{n})$ queries to the input data, where $s \leq n$ is the sparsity of the $n \times n$ input matrices $A^{(j)}$ (see, e.g., [48, 45]). This represents a polynomial speedup in the per-iteration cost compared to classical methods.

Additionally, there is also an appealing possibility that, for specific cases, the Gibbs sampling step for the $\log_2(n)$-qubit system could be accomplished in $\text{polylog}(n)$ time if the system thermalizes rapidly, allowing quantum algorithms based on the matrix MWU method to have faster runtime, perhaps as fast as $\text{poly}(\log(n), 1/\epsilon)$, representing an exponential speedup over their $\text{poly}(n, 1/\epsilon)$-time classical counterparts.

Caveats

One caveat is that the best outlook for quantum advantage occurs when the constraint matrices $A^{(j)}$ that appear in applications are sparse matrices (and

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

especially if they correspond to physical local Hamiltonians). However, this sparsity constraint may not be satisfied often in practice. There can in principle still be a speedup for dense matrices, but in this case, access to a large quantum random access memory might be required, which has its own caveats.

Another caveat to achieving a practically useful algorithm with either the classical or the quantum version of the MWU method is that the theoretical dependence of the runtime on the error parameter ϵ may lead to poor practical runtimes. The original quantum SDP solver based on MWU had $O(\epsilon^{-18})$ dependence [181], and this was later improved to $O(\epsilon^{-5})$ [45]. While this is technically poly$(1/\epsilon)$ scaling, the large power would likely lead the algorithm to be worse than alternatives, such as classical or quantum interior point methods which have polylog$(1/\epsilon)$ scaling, unless it is tolerable for ϵ to be essentially constant. In the case of zero-sum games, the quantum algorithm based on the MWU method has a slightly more tolerable $O(\epsilon^{-3})$ dependence.

Example use cases

- The MWU method can be used to gain an asymptotic quantum speedup in solving zero-sum games, and relatedly, solving LPs [46, 178]. This speedup is generated by Grover-like methods and does not require Gibbs sampling of quantum states. Many interesting optimization problems can be reduced to an LP.
- The MWU method was used in [686] to give an algorithm for online portfolio optimization, relevant to applications in finance.
- The matrix MWU method can be used to gain an asymptotic speedup for solving SDPs in the regime where the precision parameter ϵ to which the program should be optimized is large. Many interesting optimization problems can be reduced to an SDP. One notable example is that approximate solutions to (discrete) binary optimization problems can be found by solving the (continuous) SDP relaxation of the problem and performing a rounding procedure on the solution (see, e.g., [183, 71]).

Further reading

- See Arora, Hazan, and Kale [58] for an overview of the MWU method from a classical perspective, including its matrix generalization.
- The quantum algorithm for SDP based on the MWU method was introduced by Brandão and Svore [181]. This was improved in subsequent works [182, 48, 45]. The method was applied to the specific application of solving SDP relaxations of binary optimization problems in [183, 71], and to the specific application of computing optimal strategies of zero-sum games in [46].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- In [178], a "dynamic Gibbs sampling" method is proposed to improve the complexity of the MWU algorithm for zero-sum games. It would be interesting if this method can be extended to other applications of the MWU method.
- For an overview of multiplicative weights methods within quantum algorithms, see [520].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

24

Approximate tensor network contraction

The authors are grateful to Glen Evenbly, Johnnie Gray, Daniel Malz, and Ashley Milsted for reviewing this chapter.

Rough overview (in words)

Tensor network algorithms are a versatile tool that is playing an increasingly important role in problems both within and outside of physics and quantum computation [150], whenever the size of the underlying linear space is exponentially large in some appropriately defined dimension (i.e., tensor decomposition of the space). Their application to exponentially large linear systems is ultimately limited by the ability to contract (i.e., sum over repeated indices) large networks of tensors, in particular when the network forms a graph with many loops. Quantum approximate contraction of tensor networks [54] is a quantum algorithm for contracting arbitrary tensor networks up to a constant additive error. Estimating partition functions up to an additive error is a special case of the general problem, where all elements of the tensor network are positive.

This quantum approach to approximate tensor network contraction is of particular interest since many commercially relevant problems do not care about asymptotic speedups, but rather time-to-solution on smaller or medium problem sizes, and oftentimes approximate solutions found with heuristics are good enough. Tensor network (sometimes called quantum-inspired) algorithms for industrially relevant problems can be used heuristically, and the quantum approximate contraction backend might be used in cases where the classical algorithms do not provide sufficient accuracy, speed, or scale. Quantum-inspired classical algorithms based on tensor networks might allow for the identification of promising heuristic applications of quantum computing.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

At this time, however, the only known problems where the quantum backend provides substantial speedup is for problems originating from quantum computing itself, such as quantum computational supremacy experiments based on random quantum circuits [826].

Rough overview (in math)

We define a tensor network as an abstract object $T(G, M)$ defined on a graph $G = (V, E)$, where to each vertex $v \in V$ we associate a tensor $M^{(v)}$ with one index for each adjacent edge. The tensor network $T(G, M)$ is closed, in that all edges are contracted. This means that for any specific set of tensors M on G, $T(G, M)$ maps to a scalar. Given the graph G, we define a contraction pathway as a sequence of edges in G, where at each step the pair of vertices adjacent to the edge are merged into one vertex, until, at the end of the sequence, only one vertex remains in the graph. When merging two vertices, it can occur that edges that were previously distinct now coincide. It is important to keep track of the multiplicity of such edges. The optimal contraction pathway, in terms of cost scaling, is the merge sequence for which the maximum number of edges (counting multiplicity) incident to any vertex at any of the steps—a quantity called the contraction width—is minimized. The optimal contraction width is related to the tree width of the graph [741]. Classical exact contraction algorithms typically scale exponentially in the contraction width [330, 911, 483, 455]. For generic, loopy networks, the contraction width is expected to be polynomially related to $|V|$; thus, the exact contraction algorithm will quickly become intractable with growing $|V|$. However, many approximate contraction methods exist [816, 454].

Suppose we fix a contraction pathway for which the sequence of edges forms a path in G, known as a "bubbling" because one can draw a bubble around contracted vertices, which sequentially expands as the path is traversed. This may not be the pathway with optimal contraction width (for which the sequence of edges need not always form a path). Then, for any $\epsilon > 0$, there exists a quantum algorithm that runs in $O(|V|\epsilon^{-2}\text{poly}(q^d))$ quantum time and outputs a complex number r such that [54]

$$\Pr(|T(G, M) - r| \geq \epsilon\Delta) \leq \frac{1}{4},$$

where d is the maximum degree of the graph and q is the dimension of the edge Hilbert space (or bond dimension). The parameter Δ is the sequential norm of the operations in the contraction path: $\Delta = \prod_{v\in V}\|O_v\|$, where O_v are called swallowing operators (see Definitions 3.1 and 3.2 in [54]), which control the sequential contraction of the tensor network.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Intuitively, one can think of contracting the network one edge at a time along a connected path, such as a snake covering a 2D lattice. At each step of the way, the contracted vertices—which form a potentially large tensor—are encoded as a quantum state, and each new vertex is contracted by a local operator O_v (the process is called bubbling in [54]). The dimension of the "state" can increase or decrease with every operation. Each operator O_v in the contraction pathway is approximately mapped onto a unitary operator on the linear space (q^d dimensional) connecting vertex v in the network plus one ancilla qubit. The approximation comes from the Solovay–Kitaev theorem. This way, the exact contraction of the tensor network is approximately mapped onto a quantum circuit of volume roughly equal to the graph "volume." The output state of the quantum circuit encodes the result of the tensor network contraction into one of its amplitudes. In [54], they show how to estimate this amplitude using the Hadamard test, contributing the factor of ϵ^{-2} in the runtime. Alternatively, using the amplitude estimation subroutine, the ϵ-dependence could be reduced to $O(\epsilon^{-1})$.

The algorithm can be thought of as the reverse process of mapping a quantum circuit to a tensor network.

Dominant resource cost (gates/qubits)

The dominant cost of the algorithm is, on the one hand, the poly(q^d) scaling, which can be substantial for highly connected graphs. More importantly though, for problems of interest is the value of Δ, which can grow exponentially with $|V|$ and require extremely high precision ϵ to give a meaningful answer. In other words, Δ sets the scale of the approximation.

The complexity of the quantum algorithm depends sensitively on the structure of the graph $G(V, E)$, on the tensors $\{M_v\}_{v\in V}$ and on the choice of the contraction pathway. A number of limiting cases are known [54]:

- There are tensor networks for which it is NP-hard to obtain a classical additive approximation of the full contraction, suggesting the classical hardness of the problem.
- There exist families of tensor networks for which the additive approximation in Eq. (24.1) is BQP-hard, suggesting that there exists a complexity separation between the classical and quantum problem.
- There are specific examples of tensor networks representing partition functions, for which the quantum approximation scale Δ is exponential in $|V|$, but with a smaller exponent than the best known classical additive approximation scheme. There exist other examples where the converse is true [54].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Furthermore, approximate contraction of a tensor network representing a quantum partition function of a positive semidefinite Hamiltonian has been shown to be complete for the one clean qubit (DQC1) model of quantum computation [290], which suggests that approximate contraction is likely classically hard, at least for certain specific instances. Classical algorithms for this problem that are not based on tensor networks have also been examined [565].

Caveats

The main caveat at present is that we do not have a good understanding of the structure of the network that allows for significant speedup on a quantum computer, due in part to the appearance of the complicated parameter Δ in the complexity statement. It is possible that the only situations where this is possible is when the tensor network can be mapped directly to a quantum circuit, without significant overhead. For example, in [616], a specific kind of tensor network called DMERA was shown to admit an exponential quantum speedup for approximate contraction because it arises from a specific kind of quantum circuit. The speedup is in the depth of the disentangling layer of the DMERA. A more critical caveat is that we do not understand when classical contraction algorithms are inefficient in practice. For instance, certain specific classes of tensor networks can be contracted efficiently [90, 1077].

Even quantum computational supremacy experiments [63], which were designed specifically to maximize the separation between quantum and classical simulation, allow for tractable tensor network simulations up to large system sizes (~ 50) and circuit depths (~ 30) [826], though these simulations become much more challenging if we allow for nonlocal gates. Another subtle point is that it is often, or even usually, not clear what magnitude to expect for the value of a contracted tensor network from just looking at tensor norms. For example, terms may have partially canceling phases. The additive error bound for the quantum algorithm might not be very helpful in such cases.

Finally, it is likely difficult to make a proper comparison between classical approximate methods (e.g., the corner transfer matrix) and the above quantum approximation schemes, as the classical and quantum approximation errors have very different origins, and the quantum algorithm cannot be simulated at scale. The quantum algorithm might thus be regarded as a new heuristic to be be tested on a case to case basis once sufficiently powerful quantum hardware is available.

Example use cases

There is an obvious case where the quantum algorithm provides an advantage, and that is if you prepare a quantum circuit, and map it onto a tensor net-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

work. Less trivial examples involve estimating partition functions of classical statistical mechanics models—although for this problem, good classical methods exist for the additive approximation [290]. Other applications involving large-scale tensor network contractions include condensed matter physics and molecular simulations. Inference problems [342] or differential equations simulation [449] might benefit from a quantum backend in some regimes, but a careful analysis has not yet been performed.

Further reading

- Pedagogical introductions to tensor networks [201, 150, 815].
- Quantum-inspired tensor network algorithms [602, 830, 390, 817].
- Complexity analysis of the quantum partition function problem [195].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

PART III

FAULT-TOLERANT QUANTUM COMPUTING

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Overview

Throughout this book, we predominantly restrict our attention to the circuit model of quantum computation. Within this paradigm, any quantum algorithm can be expressed as a sequence of basic operations, such as product state preparation, unitary single- and two-qubit gates, and single-qubit Pauli measurements. In order to accurately determine complete end-to-end resource estimates for quantum algorithms it is essential to understand the costs of (i) decomposing quantum algorithms into basic operations and (ii) realizing these basic operations reliably with the physical hardware. In other parts of this survey we assume noiseless logical qubits and operations (unless otherwise noted) and focus on item (i). In this part, we take into account that physical qubits and operations are noisy and discuss item (ii). We first review the fundamental ideas behind the theory of fault tolerance. We then illustrate them with concrete realizations in the paradigm of the surface code and lattice surgery.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

25

Basics of fault tolerance

The authors are grateful to Earl Campbell, Andrew Cross, and John Preskill for reviewing this chapter.

Rough overview (in words)

The error rates of all known realizations of physical qubits and basic operations are too high to enable implementation of the majority of quantum algorithms considered in this book. Even if the probability p for each basic operation to malfunction was minute, we would nevertheless expect an error to occur in any quantum circuit comprising more than $O(1/p)$ operations. One may optimistically assume that in the foreseeable future $p = 10^{-6}$ might be achieved by certain quantum architectures, such as trapped ions [125, 208]. This, in turn, limits the size of any quantum circuit that one may hope to reliably execute to roughly one million basic operations. Such a bound places a severe restriction on the algorithms that could be run and is orders of magnitude smaller than the resources needed to implement the quantum algorithms described in the other parts of this survey.

The theory of quantum fault tolerance [936] and quantum error correction [935, 952, 444] provides a collection of techniques to deal with imperfect operations and unavoidable noise afflicting the physical hardware, at the expense of moderately increased resource overheads. In the basic model for fault tolerance, one assumes that each elementary component of a quantum circuit (including the identity gate) may fail with some small but nonzero probability, independently of the other components, and classical information processing is noiseless. For concreteness and simplicity, one may choose to model any noisy component as an ideal component followed by (or, in the case of measurements, preceded by) some Pauli channel acting on the same subset of

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

qubits. Let C be a quantum circuit (possibly with classical input and output) describing a desired quantum algorithm. Since each component of C may fail, one should not implement C directly; rather, one needs to implement a different quantum circuit $\mathcal{F}(C)$, which is a fault-tolerant (FT) version of C. This, in turn, can be achieved by replacing each qubit in C with a logical qubit encoded in some quantum error correcting (QEC) code and each elementary component of C with a corresponding FT gadget; see Fig. 25.1. The desired quantum computation will then be realized on the logical level of $\mathcal{F}(C)$ without leaving the protective encoding guaranteed by the QEC code.

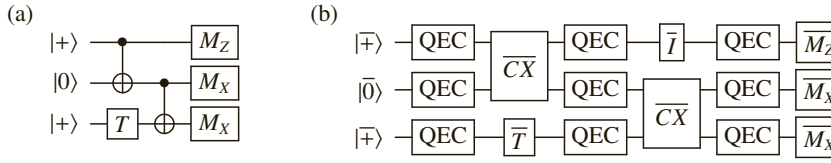


Figure 25.1 (a) A quantum circuit C consists of state preparation, unitary gates, and measurements. (b) An FT realization of C is a quantum circuit $\mathcal{F}(C)$ obtained by replacing each qubit in C with a logical qubit encoded in some QEC code and using appropriate FT gadgets interspersed with QEC gadgets in place of each basic component of C. Note that some gadgets may require considerable resources (not shown in the picture); see Chapter 26 on quantum error correction and Chapter 27 on logical gates with the surface code for more details.

To realize universal FT quantum computation, it suffices to have state preparation gadgets (for at least one type of state), measurement gadgets (for at least one type of measurement), gate gadgets (for a universal set of gates), and QEC gadgets. One requires that all of these gadgets satisfy certain FT conditions; see, for instance, [20, 446]. Although the asymptotic scaling of resource overheads associated with FT gadgets is manageable (e.g., polylogarithmic in the inverse of the target logical error rate), the constant prefactors tend to be large, resulting in the qubit and time overheads that currently constitute one of the main bottlenecks to practical FT quantum computation. We will discuss this point in more detail for the implementation of logical gates and quantum error correction with the planar architecture based on the surface code [626, 343].

Rough overview (in math)

Designing FT gadgets is a challenging task for several reasons. First, FT gadgets are usually developed and optimized for a specific QEC code. Second, even though they comprise imperfect basic components, they are required to work reliably as long as the number of malfunctioning components is limited.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Third, FT gadgets may spread errors, however, they must not do so in an uncontrollable way.

Given a set of FT gadgets, one can reliably perform an arbitrarily long quantum computation as long as the physical error rate of each basic component is below some constant value, often referred to as the FT threshold. This result is established by the celebrated threshold theorem [10, 625, 636, 20]. To be more precise, consider the basic model for FT. The threshold theorem asserts that there exists a constant $p_{\mathrm{FT}} > 0$, such that for any $\epsilon > 0$ and any quantum circuit C there exists a quantum circuit $\widetilde{C}$ that produces an output with statistical distance at most ϵ from the output of C, provided the physical error rate p is below p_{FT}. Moreover, $\widetilde{C}$ uses a number of qubits and time steps that are at most polylog$(|C|/\epsilon)$ times bigger than the number of qubits and time steps in C, where $|C|$ denotes the number of basic components in C.

The basic idea behind the proof of the threshold theorem proceeds as follows. Consider a quantum circuit $\mathcal{F}(C)$, which is an FT implementation of C. Assuming the basic model for fault tolerance described above, for sufficiently small physical error rate p, the logical error rate for $\mathcal{F}(C)$ should be smaller than p, since $\mathcal{F}(C)$ is an FT implementation of C. One can then consider a quantum circuit $\mathcal{F} \circ \mathcal{F}(C)$, which is an FT implementation of $\mathcal{F}(C)$, reducing the logical error rate even further. By repeating this process, one eventually obtains a quantum circuit $\widetilde{C} = \mathcal{F} \circ \cdots \circ \mathcal{F}(C)$ with the logical error rate below ϵ. The resulting FT protocol is based on concatenated QEC codes.

One may improve the scaling of the resource overheads from the threshold theorem with concatenated QEC codes. In particular, in the asymptotic limit of large quantum circuits, the ratio of qubits in C and $\widetilde{C}$ can be a constant [447]. In this construction, the FT protocol requires a family of QEC codes that satisfies certain properties, including the desired scaling of code parameters, computationally efficient decoding algorithms, and constant-weight parity checks. Such a family of QEC codes was first provided in [388].

Dominant resource cost (gates/qubits)

At the heart of FT quantum computation, there is usually some QEC code. The choice of a QEC code affects the relationship between a quantum circuit C and its FT realization $\widetilde{C}$, and the subsequent scaling of the resource overheads. Therefore, we would like to choose a QEC code for which the encoding rate (defined as the ratio k/n, where k and n are the number of logical and physical qubits, respectively) as well as the relative code distance (defined as the ratio d/n, where d is the minimum weight of any nontrivial logical operator) are as high as possible. Although for concatenated QEC codes (that feature in the threshold theorem), both k/n and d/n go to zero as n goes to infinity,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

we know that there exist QEC codes with good parameters, that is, for which k/n and d/n are asymptotically constant [221]. Moreover, recent groundbreaking results [199, 827, 352, 678] provided constructions of QEC codes that not only have good parameters but also constant-weight parity checks (thus their name—quantum low-density parity check codes). The latter property is particularly important from the perspective of fault tolerance. In fact, all quantum low-density parity check codes with code distance $d = O(\log n)$ have a nonzero FT threshold [643]. However, experimental realization of these constructions (in contrast to the surface code) seems to become extremely challenging as the number of physical qubits increases, at least within the realm of solid-state qubits constrained by geometric locality of their physical entangling gates.

Another aspect of FT quantum computation that affects the resource overheads are the FT gadgets being used. One of the easiest ways to implement FT gadgets for gates is via transversal gates. By definition, transversal gates are implemented via a tensor product of single-qubit unitaries (or, more generally, via a depth-one quantum circuit) and therefore do not spread errors in an uncontrollable way. Unfortunately, transversal gates are limited by the Eastin–Knill theorem [370, 1078, 583, 650], which rules out the existence of a (finite-dimensional) QEC code with a universal set of transversal logical gates. One strategy to circumvent this limitation is to prepare certain magic states and use them to realize FT gates [190]; see Chapter 27 on implementing logical gates for more details and a discussion of other strategies.

To realize FT gadgets for state preparation, QEC, and measurement, one typically chooses among three FT schemes: Shor's [936], Steane's [953], or Knill's [634]. Roughly speaking, Shor's scheme uses simple states (verified cat states) of the ancilla qubits at the expense of implementing many gates on the data qubits, whereas Steane's and Knill's schemes trade highly complex states of the ancilla qubits (logical states encoded in the underlying QEC code) for minimizing the number of gates on the data qubits. To determine the best choice, one needs to consider the underlying QEC code (e.g., Steane's scheme is applicable only to the subset of codes known as CSS codes [221, 952]) and the quantum hardware restrictions (e.g., lack of extra ancilla qubits). For an illuminating and detailed discussion of FT schemes, see [446]. There are FT schemes that do not squarely fit in the aforementioned classification, for example, [353, 254, 544]. Also, for QEC codes with additional structure, such as quantum low-density parity check codes, one may pursue different approaches toward FT quantum computation; see Chapter 26 on QEC with the surface code.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Caveats

Rigorous proofs provide lower bounds on the FT threshold p_{FT}. For instance, for an FT scheme based on the 7-qubit code, one finds $p_{\text{FT}} > 2.73 \times 10^{-5}$ [20]. For an FT scheme by Knill [634] that relies on complex ancilla preparation techniques, one finds $p_{\text{FT}} > 1.04 \times 10^{-3}$ [21]. However, these values can differ by orders of magnitude from the values estimated in numerical simulations. For instance, the FT scheme by Knill is estimated to have an FT threshold p_{FT} as high as 5×10^{-2}, constituting one of the highest-known FT thresholds. We remark that these values depend sensitively on the details of the FT schemes and the assumptions about noise. In particular, to obtain the aforementioned values we assume the ability to implement gates between any qubits. On the other hand, if we arrange qubits on a geometric lattice and restrict gates to be local, then while FT thresholds still exist, their values are significantly reduced.

One can expand the threshold theorem in many ways. Even using the basic model for fault tolerance, one may choose the failure probabilities for each elementary component of a quantum circuit differently, for example, the failure probability of a measurement to be an order of magnitude higher than that of a gate. One can consider more general noise (which includes systematic errors, such as overrotations) arising due to a weak interaction between the system and a non-Markovian environment [20, 986]. In general, although experimental realizations of quantum computation may not satisfy exactly the assumptions of the threshold theorem, we expect the main conclusions to hold as long as the assumptions are not violated too much.

To simplify the analysis of FT schemes, we often assume unlimited classical computational power that one needs to, for example, process the error syndrome and infer an appropriate recovery operator in a QEC gadget; a number of such decoding algorithms have been developed for QEC with the surface code. It is important not to abuse this assumption by, for instance, solving the initial problem with an inefficient classical algorithm. At some point, however, one needs to take into account the finite speed of classical information processing. If the classical unit that processes the error syndrome is unable to keep pace with the rate at which this syndrome is being produced, then the error syndrome will start to accumulate and one will suffer from the so-called backlog problem [985]. Subsequently, the speed of quantum computing will be exponentially reduced and the computational advantage of quantum computing will be annulled. This issue will be especially prominent for quantum algorithms with only polynomial speedups.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Further reading

- An accessible introduction to quantum error correction and the theory of fault tolerance can be found in [861].
- A detailed introduction to quantum error correction and fault-tolerant quantum computation can be found in [446].
- A perspective on roads toward fault-tolerant universal quantum computation can be found in [226].
- The error correction zoo [17] provides a catalog of error correcting codes.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

26

Quantum error correction with the surface code

The authors are grateful to Earl Campbell, Andrew Cross, and John Preskill for reviewing this chapter.

Rough overview (in words)

To protect quantum information from the detrimental effects of noise, we can encode it into a code space of some quantum error correcting (QEC) code [935, 952]. Oftentimes, we choose to work with stabilizer codes [444]. By definition, a code space of a stabilizer code is the simultaneous (+1)-eigenspace of a set of commuting Pauli operators, commonly referred to as parity checks.

The surface code [626, 196, 343] is one of the most studied stabilizer codes. It can be implemented with a planar layout of qubits and entangling gates only between neighboring qubits. For that reason, the surface code is particularly appealing for quantum hardware architectures with restricted qubit layout and connectivity, such as superconducting circuits [348, 155]. The most common realization of the surface code uses $n = L^2$ data qubits to encode $k = 1$ logical qubit and has code distance $d = L$, where L is the linear size of the $L \times L$ square lattice with open boundary conditions. Additionally, $n_A = L^2 - 1$ ancilla qubits are typically used to measure parity checks; see Fig. 26.1(a).

In order to perform QEC, we have to be able to detect errors without revealing the encoded information. For stabilizer codes, we can achieve that by measuring their parity checks to obtain the error syndrome (which comprises the measurement outcomes returning -1). Then, the error syndrome is processed by specialized classical algorithms, also known as "decoders," to find an appropriate recovery operator that attempts to remove errors afflicting the encoded information. For generic stabilizer codes, the problem of optimal decoding is computationally hard, even for simple noise models [563]. However,

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

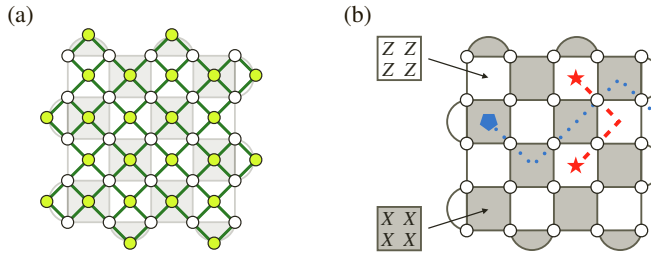


Figure 26.1 (a) A planar layout of data and ancilla qubits (empty and filled dots, respectively) with entangling gates (thick edges) only between neighboring qubits. This layout gives rise to the $L \times L$ square lattice with open boundary conditions, where $L = 5$ here. (b) The surface code can be realized by measuring Pauli Z- and X-type parity checks (light and dark faces, respectively). The error syndrome (stars and pentagons) can be interpreted as the endpoints of string-like Pauli X and Z errors (dashed and dotted lines, respectively).

for QEC codes with some underlying structure, such as the surface code, there exist a variety of computationally efficient (albeit not optimal) decoding algorithms. In particular, the three most popular classes of decoders for the surface code are as follows:

- Matching decoders, including the minimum-weight perfect matching algorithm [343] and its follow-up improvements, such as the belief-matching algorithm [320, 530]. These decoders phrase the problem of surface code decoding as a graph-theoretic problem of perfect matching, which can be efficiently solved [372].
- Clustering decoders, such as the renormalization-group decoder [364, 42] and the union-find decoder [338]. These decoders primarily exploit the structure of the error syndrome in the surface code; see Fig. 26.1(b).
- Tensor network decoders [192, 331, 291]. These decoders phrase the problem of surface code decoding as a numerical problem of contracting tensor networks.

In order to assess the usefulness of decoders, one usually considers two criteria: runtime and accuracy. The first criterion, runtime, is defined as the time needed for the decoder to process the error syndrome. It is crucial that any practical decoder is able to operate at a rate compatible with the rate of parity check measurements; otherwise, the error syndrome will start to accumulate, leading to the backlog problem [985]. The second criterion, accuracy, is typically defined for a given noise model in terms of the logical error rate, that is, the failure rate of the decoder to successfully undo the effects of noise on

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the encoded information. From the perspective of reducing runtime and improving accuracy, matching and clustering decoders stand out. Namely, they can achieve almost-linear runtime [529, 338] and good accuracy. Combined with the techniques of parallelizable real-time decoding [942, 170], these decoders will likely play a key role in scalable QEC with the surface code. To achieve optimal accuracy, one can use tensor network decoders, however, they are often not computationally efficient, with runtime that scales unfavorably.

Rough overview (in math)

In addition to being compatible with planar layouts of qubits and admitting computationally efficient decoders with good accuracy, the surface code also exhibits one of the highest QEC thresholds. Recall that a QEC threshold is specified for the following triple: a QEC code family of growing distance d, a decoder, and a noise model. It is defined as the highest value p_{th} such that for any error rate $p < p_{\text{th}}$ the probability that the decoder fails to undo the effects of noise goes to zero as d goes to infinity. For example, the QEC threshold for the surface code, using the minimum-weight perfect matching algorithm for decoding, with a circuit noise model based on depolarizing noise, is around 1% [1017, 530].

Typically, if the error rate p describing noise is sufficiently low and below the threshold p_{th}, then the logical error rate p_{fail} roughly scales as

$$p_{\text{fail}} \sim \left(\frac{p}{p_{\text{th}}}\right)^{\lceil \frac{d}{2} \rceil}. \tag{26.1}$$

This implies that in order to achieve the target error rate ϵ, it suffices to implement the surface code with code distance $d = O(\log(1/\epsilon)/\log(p_{\text{th}}/p))$ using $n + n_A = O(d^2) = O(\log^2(1/\epsilon)/\log^2(p_{\text{th}}/p))$ data and ancilla qubits. Subsequently, qubit overhead associated with QEC based on the surface code scales polylogarithmically in the inverse target error rate $1/\epsilon$.

Dominant resource cost (gates/qubits)

Performing reliable QEC in the presence of measurement errors becomes challenging since the error syndrome can be corrupted. A straightforward solution to the problem of unreliable error syndrome is to repeatedly measure the parity checks in order to gain enough confidence in their measurement outcomes [936, 343]. If this approach is applied to the surface code with code distance d, then one needs to perform $O(d)$ rounds of parity check measurements, incurring relatively large time overhead.

To reduce time overhead, one can pursue single-shot QEC [166], which does not require repeated measurement rounds. It is possible to realize single-shot

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

QEC with the surface code [224, 65, 339], however, in addition to the parity checks in Fig. 26.1(b), one would need to measure nonlocal high-weight parity checks, which is a serious limitation. A more streamlined approach is to consider a different realization of the surface code, the 3D subsystem toric code [652, 202], which can be implemented with qubits arranged on the cubic lattice and local low-weight parity checks. Although this approach is natively defined in three spatial dimensions, it can be emulated with planar layouts of qubits and either a limited number of nonlocal gates or the ability to reshuffle qubits (which is available with, e.g., Rydberg atoms [890, 203]). In order to realize code distance d, one incurs qubit overhead of $O(d^3)$ (compared to qubit overhead of $O(d^2)$ for the surface code). From that perspective, single-shot QEC with the subsystem toric code can be viewed as trading time overhead for qubit overhead.

Caveats

There have been efforts to improve surface code decoders by incorporating various machine learning methods, including neural networks [992, 746, 251, 1003, 107] and reinforcement learning [973]. At the current stage, decoders solely based on machine learning methods seem to be of limited applicability, mostly due to high training costs and scalability issues. Nevertheless, these approaches are likely to be immensely beneficial for QEC in the settings where (possibly correlated) noise is unknown and may have to be learned first.

Typically, in QEC analysis one considers simple Pauli noise, such as depolarizing noise acting independently and identically on each qubit. If noise exhibits bias between the X, Y, and Z components of Pauli noise, then this structure can be exploited, leading to dramatically increased QEC thresholds, as exemplified by variants of the surface code [999, 173, 363]. Similarly, noise that is biased toward erasure errors can be beneficial from the perspective of QEC [947, 1057, 654, 467]. On the other hand, realistic noise may be coherent or correlated and thus not only difficult to correct, but also to numerically simulate. For instance, the logical error rates for coherent noise may be orders of magnitude higher than the estimates of the logical error rates for simple Pauli noise (assuming both types of noise have the same error rate) [564]. In certain cases of coherent noise, however, QEC with the surface code may be efficiently simulable [193].

In addition to the 3D subsystem toric code, one can also consider other higher-dimensional versions of the surface code. With these codes, roughly speaking, one improves the QEC capabilities at the expense of increased qubit overhead. Moreover, for the higher-dimensional surface code, it may suffice to use arguably the least complex decoders that are based on cellular automata

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

(which, by definition, are parallelizable and only use local information about the error syndrome) [343, 200, 651, 1005].

Example use cases

- Decoders for the surface code can be used for other QEC code families, such as the color code [169, 168, 647]. In fact, due to a close connection between the color codes and the surface codes [171, 653], any surface code decoder can be used as a subroutine in the restriction decoder for any color code (in two or more spatial dimensions) [649, 1004].

Further reading

- The seminal paper by Dennis et al. [343] is a thorough introduction to QEC with the surface code.
- A recent perspective [205] discusses how to use matching decoders to decode stabilizer codes.
- Open-source software packages have been developed for implementing QEC with the surface code, such as Stim [422] and PyMatching [528].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

27

Logical gates with the surface code

The authors are grateful to Earl Campbell, Andrew Cross, and John Preskill for reviewing this chapter.

Rough overview (in words)

The ability to implement an arbitrary unitary operation, either exactly or approximately, is a prerequisite for performing quantum computation. It can be achieved with unitary gates that form a universal gate set [625, 801]. A commonly considered gate set contains two Clifford gates, the Hadamard gate H and the controlled X gate CX (also known as the controlled NOT gate), and one non-Clifford gate, the $T = Z^{1/4}$ gate. One can consider other non-Clifford gates, such as the Toffoli gate CCX. Note that non-Clifford gates are essential for quantum computation, as any quantum circuit comprising only Clifford gates, state preparation, and measurement in the computational basis can be simulated in polynomial time on a probabilistic classical computer [445, 4].

Since we are interested in fault-tolerant quantum computation, we would like to implement a universal set of logical gates $\overline{H}$, $\overline{CX}$, and $\overline{T}$ on information encoded in some quantum error correcting (QEC) code, such as the surface code. We can implement these gates with a planar layout of qubits and nearest-neighbor entangling gates. To be more precise, we consider a simple architecture [535] that comprises N surface code patches, each encoding a logical qubit into the surface code with code distance d, and the routing space in between; see Fig. 27.1(a). In such an architecture, the total number of data and ancilla qubits is $O(Nd^2)$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

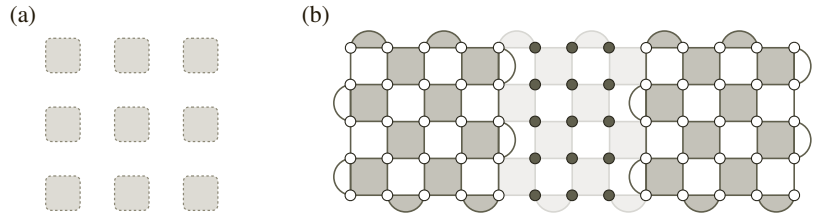
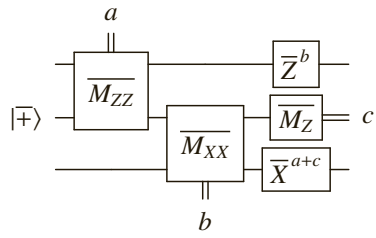


Figure 27.1 (a) A planar layout of qubits comprises surface code patches (shaded), each using the layout depicted in Fig. 26.1(a) and encoding a logical qubit, with routing space in between the patches. (b) The logical Pauli measurement $\overline{M_{XX}}$ is implemented by preparing the routing space qubits (filled dots) in the state $|0\rangle$ and repeatedly measuring parity checks (lightly shaded) in the routing space spanning between the two surface code patches. Other logical Pauli measurements, for example, $\overline{M_{ZZ}}$ and $\overline{M_{YZ}}$, require connecting different boundaries of the two patches.

Rough overview (in math)

The logical $\overline{H}$ does not pose any challenges. From a practical standpoint, it is transversal, since it can be realized by applying the Hadamard gate H to every data qubit in the surface code patch, followed by swapping of the roles of Pauli Z- and X-type parity checks in the subsequent QEC rounds. As such, the logical $\overline{H}$ takes constant time and the surface code patch is effectively rotated (which may alter how subsequent operations are implemented).

The logical $\overline{CX}$ is more challenging than the logical $\overline{H}$, since it is impossible to implement it transversally with the planar layout of qubits and nearest-neighbor entangling gates shown in Fig. 27.1(a). Instead, one can use the following quantum circuit, where the first qubit (top wire) is the control and the third qubit (bottom wire) is the target of the logical $\overline{CX}$ gate:



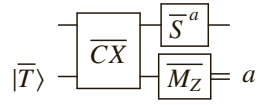
(27.1)

It is straightforward to fault-tolerantly realize preparation of the logical state $|\overline{+}\rangle$, logical Pauli measurement $\overline{M_Z}$, and logical Pauli operators $\overline{Z}$ and $\overline{X}$. In addition, the required logical Pauli measurements $\overline{M_{ZZ}}$ and $\overline{M_{XX}}$ can be implemented fault-tolerantly via "lattice surgery" techniques [535, 397, 693]; see Fig. 27.1(b) for an illustration of how to realize $\overline{M_{XX}}$. Unlike the logical

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

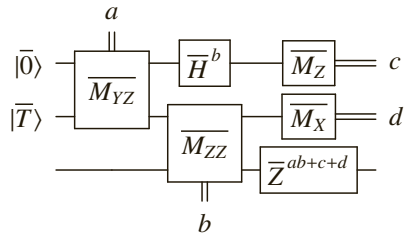
$\overline{H}$, logical Pauli measurements $\overline{M_{ZZ}}$ and $\overline{M_{XX}}$ and, subsequently, the logical $\overline{CX}$ cannot be realized in constant time; rather, due to the need to account for measurement errors, they typically incur time overhead of $O(d)$.

The logical $\overline{T}$ can be implemented using gate teleportation [448] via the following quantum circuit:



(27.2)

Here, the logical resource state $|\overline{T}\rangle$ is defined as $\left(|\overline{0}\rangle + e^{i\pi/4}|\overline{1}\rangle\right)/\sqrt{2}$, the logical gate $\overline{S}$ is defined as $\overline{Z}^{1/2}$, and the first qubit (top wire) is the control and the second qubit (bottom wire) is the target of the logical $\overline{CX}$ gate. One can fault-tolerantly implement the logical $\overline{S}$ with a planar layout of qubits [206, 423] (or even in a transversal way given access to nonlocal entangling gates [653, 788]). However, the need to apply the logical $\overline{S}$ conditioned on the measurement outcome of $\overline{M_Z}$ may slow down quantum computation, and, for this reason, it may be beneficial to use the following quantum circuit from [693, Fig. 17(b)], which is an alternative to the one in Eq. (27.2) that uses one additional logical qubit but requires only logical Pauli corrections, rather than logical Clifford corrections.



(27.3)

In either case, given the logical resource state $|\overline{T}\rangle$, the logical $\overline{T}$ typically incurs time overhead of $O(d)$. We conclude that implementing the logical $\overline{T}$ reduces to the problem of preparing the logical state $|\overline{T}\rangle$, which, in turn, can be realized via state distillation [633, 190]; see [146] for a brief overview of state distillation.

Dominant resource cost (gates/qubits)

State distillation provides a fault-tolerant method to prepare high-fidelity logical resource states, such as the logical state $|\overline{T}\rangle$. The basic idea is to convert some number of noisy resource states into fewer but, crucially, less noisy resource states. Importantly, this task can be accomplished with quantum circuits comprising only Clifford gates (together with state preparation and measurement in the computational basis) and postselection. Typically, state distillation

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

circuits are based on some QEC code, for example, the 15-qubit Reed–Muller code.

State distillation is often described as a resource-intensive method that contributes the most to the resource overhead of fault-tolerant quantum computation with the surface code [398] (assuming many state distillation circuits working in parallel). For that reason, numerous efforts have been devoted to finding possible alternatives [189, 582, 167, 249, 146]. However, recent results indicate that state distillation may not be as costly as one may think [693, 694], especially when one allows only a few state distillation circuits to run in parallel and optimizes them for specific quantum hardware and noise that exhibits some bias [696]. In the task of estimating the ground state energy density of the Fermi–Hubbard model, state distillation of logical Toffoli resource states injected one at a time uses less than 10% of the total resources and is never a bottleneck on runtime of the quantum algorithm [250].

Often, a quantum algorithm is expressed as a quantum circuit C comprising Clifford and T gates. Thus, by using the aforementioned logical gates $\overline{H}$, $\overline{CX}$, and $\overline{T}$, we can fault-tolerantly implement the logical quantum circuit $\overline{C}$ with the surface code of code distance d and a planar layout of qubits in Fig. 27.1(a). However, from the perspective of reducing the resource overheads, it may be beneficial to consider a quantum circuit C' equivalent to the circuit C, which is obtained from C by commuting all Clifford gates to the end of C [693]. As a result, the circuit C' only comprises multiqubit Pauli $\pi/8$ rotations (which are a generalization of the T gate and can be realized via, e.g., quantum circuits analogous to the one in Eq. (27.3)). Consequently, fault-tolerant implementation of the logical circuit $\overline{C'}$ incurs qubit overhead of $O(Nd^2)$ and time overhead of $O(Md)$, where N and M are the number of qubits and T gates in C, respectively. We remark that the time overhead can be reduced at the expense of increased qubit overhead—first, by distilling more resource states and being able to use them faster, then, by implementing them in parallel [693].

Caveats

Lattice surgery is not necessary to realize fault-tolerant quantum computation with a planar layout of qubits and nearest-neighbor gates. An alternative approach (which actually preceded the development of lattice surgery) relies on the surface code with defects and braiding [862, 863, 398, 206]. However, resource overhead estimates strongly suggest that this approach is not competitive with lattice surgery [397].

The simple architecture depicted in Fig. 27.1(a) can be improved in a couple of ways to reduce the qubit overhead. First, it is possible to pack surface code patches more densely, resulting in more logical qubits for the given total

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

number of qubits and target code distance [660, 693]. Second, one can designate certain regions, commonly referred to as magic state factories, to solely produce resource states, such as the logical state $|\overline{T}\rangle$, and optimize their design [809, 693, 694].

To simplify implementation of logical gates, one can consider other QEC codes, for example, the 3D color code [165, 648]. The gauge color code has redundant degrees of freedom, commonly referred to as gauge qubits. For different states of its gauge qubits, the gauge color code admits transversal implementation of different logical gates, which, *combined*, form a universal gate set (thus circumventing the Eastin–Knill theorem [370, 1078]). Importantly, changing the state of gauge qubits can be done fault tolerantly in constant time. However, to realize this construction one needs, for instance, a 3D layout of qubits with nearest-neighbor gates or a planar layout of qubits with a limited number of nonlocal gates, which are more challenging to engineer compared to the simple architecture in Fig. 27.1(a). To achieve code distance d with the gauge color code, one incurs qubit overhead of $O(d^3)$ (compared to qubit overhead of $O(d^2)$ for the surface code), so, similarly to single-shot QEC described in Chapter 26 on QEC with the surface code, this approach trades time overhead for qubit overhead.

Example use cases

- Lattice surgery techniques developed for the surface code can be straightforwardly adapted to, for example, the color code [659] or the surface code with a twist [1066], leading to fault-tolerant quantum computation with potentially reduced qubit overhead. Recent work [303] proposed a generalization to the setting of quantum low-density parity check codes. In addition, lattice surgery techniques can also be used for the fault-tolerant transfer of encoded information between arbitrary topological quantum codes [842].
- Now, we are ready to present a rough, order-of-magnitude estimate of the resource overheads needed to realize fault-tolerant quantum computation in the architecture based on the surface code and lattice surgery. For concreteness, we consider the circuit noise of strength $p = 0.001$, where each basic operation, including state preparation, CNOT gate, and measurement, can fail with probability p. Assume that we want to implement a quantum circuit C comprising $N = 10^3$ qubits and a certain number $M = 10^{10}$ of T gates. These resource counts are in the ballpark of estimates for various quantum algorithms in the application areas of quantum chemistry, condensed matter physics, and cryptanalysis. First, following the procedure from [693], we compile C into a new circuit C' of depth M that comprises N qubits and M multiqubit Pauli $\pi/8$-rotations implemented one at a time. Since there are

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

NM possible fault locations in the circuit C', the error rate for each qubit of C' should not exceed

$$\epsilon \approx 1/(NM).$$

Since each qubit of C' is realized as a logical qubit of the surface code with distance d, then its logical error rate p_{fail} can be approximated by

$$p_{\text{fail}} \approx \alpha(p/p_{\text{th}})^{d/2},$$

where we can crudely set $\alpha = 0.05$ and $p_{\text{th}} = 0.01$; see Chapter 26 on QEC with the surface code for more details. Note that these values are empirical and depend heavily on the choice of the decoder, in our case, the belief-matching algorithm [530]. Thus, in order for the logical error rate p_{fail} to reach the target error rate ϵ we need the surface code distance at least

$$d \approx \lceil 2\log(\alpha NM)/\log(p_{\text{th}}/p)\rceil.$$

Assuming that half of all required qubits are devoted to realizing N surface code patches (each comprising $2d^2 - 1$ data and ancilla qubits), with the other half used for resource state distillation and routing [693], we obtain that the fault-tolerant implementation of C' incurs qubit overhead of

$$n_{C'} \approx 4Nd^2$$

and time overhead of

$$t_{C'} \approx Md\tau,$$

where we crudely set $\tau = 1\mu s$ to be the time needed to implement one syndrome measurement round with the superconducting circuits architecture. Finally, our order-of-magnitude resource estimate gives 2.3×10^6 physical qubits and 67 hours of runtime. This general approach to resource estimation has been applied to a number of specific quantum algorithms in a variety of application areas; see, for example, [669, 424, 630, 147, 895]. These references often go beyond a back-of-the-envelope calculation and provide a more meticulous analysis that accounts for exact qubit layouts and the physical footprint of resource state distillation factories. They also pursue optimizations to how the circuit is implemented (e.g., exploiting space-time tradeoffs) in light of these considerations.

Further reading

- An accessible overview of fault-tolerant quantum computation based on the surface code and lattice surgery can be found in [693].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

- A convenient way to describe and optimize lattice surgery operations is via the ZX calculus, which is a diagrammatic language for quantum computing [302, 335].
- A direct comparison of the resource overhead associated with preparation of the logical resource state $|\overline{T}\rangle$ using either state distillation or transversal gates (with the 3D color code) can be found in [146].
- To read about a framework for estimating resources required to realize large-scale fault-tolerant quantum computation, see [147].
- The recently introduced paradigm of algorithmic fault tolerance [1090] may significantly reduce the space-time overhead of FT quantum computation with the surface code.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Appendix

Background, conventions, and notation

This book aims to be as modular as possible, where each numbered chapter and section can be read on its own—as such, specific notational definitions, technical terms, and acronyms are redefined the first time they appear in each numbered section.

Nevertheless, the mathematical presentation throughout this book does assume familiarity with certain concepts, techniques, and conventions that are ubiquitous within the field of theoretical quantum information science. This book is targeting an interdisciplinary community of researchers; thus, the assumed conventions and understanding of "common knowledge" will vary from reader to reader, depending on one's background and experience. In order to make the material as widely accessible as possible, here we collect some of the concepts and notational choices that are commonly used throughout this book.

For readers interested in a more complete introduction to the field and its standard conventions, we recommend the following resources:

- The definitive reference in the field of quantum computation is the book by Nielsen and Chuang [801]. Other classic textbooks include those by Kitaev, Shen, and Vyalyi [627] and Kaye, Laflamme, and Mosca [606]. A similarly general set of topics is covered in the lecture notes of Preskill [845].
- Several sets of more recent lecture notes have a specific focus on quantum algorithms, for example, by de Wolf [1051], Childs [276], and Lin [687]. See also the review article on quantum algorithms by Montanaro [774].
- Some online resources include a website containing lecture notes on quantum algorithms for data analysis and quantum machine learning by Luongo [729], the Pennylane codebook [123], and the quantum algorithm zoo [586].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

A.1 Quantum systems and bra-ket notation

Basic concepts from linear algebra are an essential prerequisite for understanding quantum computation and thus for much of the technical discussion in this book. We adopt bra-ket notation to denote quantum states and the linear algebraic objects they correspond to. The state of a quantum system, such as a collection of qubits, is labeled by a *ket*, such as $|\psi\rangle$—this object corresponds to a vector in a finite-dimensional vector space. For instance, the state of a single qubit is an element of the 2D complex vector space $\mathbb{C}^2$, and can be represented by a length-2 column vector or by a *superposition* over orthonormal basis states, denoted $|0\rangle$ and $|1\rangle$:

$$|\psi\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = \alpha_0|0\rangle + \alpha_1|1\rangle, \qquad \alpha_0, \alpha_1 \in \mathbb{C}\,.$$

The state of a system of n qubits is an element of the 2^n-dimensional complex vector space $\mathbb{C}^{2^n}$—the tensor product of the n individual 2D vector spaces—and can be represented in any of the following equivalent ways:

$$|\psi\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{2^n-1} \end{pmatrix} = \sum_{x=0}^{2^n-1} \alpha_x|x\rangle = \sum_{x=0}^{2^n-1} \alpha_x|x_0\rangle \otimes |x_1\rangle \otimes \cdots \otimes |x_n\rangle, \qquad \alpha_x \in \mathbb{C}\ \forall x\,,$$

where $x_i \in \{0, 1\}$ denotes the i-th bit of the integer x when x is written in binary (leading zeros are added such that x has n digits, and x_0 corresponds to the most significant bit). The orthonormal basis $|0\rangle, |1\rangle, \ldots, |2^n - 1\rangle$ is called the *computational basis*. An n-qubit quantum state is said to be a *product state* if it can be written as a tensor product of 2D states on each of the n systems

$$|\psi\rangle = |\phi_1\rangle \otimes |\phi_2\rangle \otimes \cdots \otimes |\phi_n\rangle, \qquad |\phi_i\rangle \in \mathbb{C}^2\ \forall i\,,$$

and it is said to be *entangled* if it cannot be written as a product state.

For each quantum state $|\psi\rangle$ corresponding to a column vector as above, we denote its Hermitian adjoint (i.e., the complex conjugate of its transpose) by the *bra* $\langle\psi|$, which corresponds to the row vector

$$\langle\psi| = \begin{pmatrix} \alpha_0^* & \alpha_1^* & \cdots & \alpha_{2^n-1}^* \end{pmatrix},$$

where α_x^* denotes the complex conjugate of α_x. A bra $\langle\phi| = \sum_x \beta_x^* \langle x|$ and ket $|\psi\rangle = \sum_x \alpha_x|x\rangle$ together form a braket

$$\langle\phi|\psi\rangle = \sum_{x=0}^{2^n-1} \beta_x^* \alpha_x \in \mathbb{C}\,,$$

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

which is simply the standard Hermitian inner product between vectors $|\phi\rangle$ and $|\psi\rangle$. The *norm* of the state $|\psi\rangle = \sum_x \alpha_x |x\rangle$ refers to the standard Euclidean vector norm, or 2-norm of the vector, given by

$$\| |\psi\rangle \| = \sqrt{\langle\psi|\psi\rangle} = \sqrt{\sum_{x=0}^{2^n-1} |\alpha_x|^2}\,.$$

A state for which $\| |\psi\rangle \| = 1$ is said to be *normalized*; in this book, kets are usually (but not always) normalized.

The above corresponds to the case for *pure* quantum states; in some instances, we consider the more general case that the state of the quantum system is *mixed*—that is, it is drawn from a probabilistic ensemble of multiple pure quantum states. In this case, an n-qubit quantum state is represented by a $2^n \times 2^n$ matrix called a *density matrix*, typically denoted by a lowercase Greek letter such as ρ. A matrix ρ is a valid quantum state if it is Hermitian and positive semidefinite. Furthermore, it is a normalized quantum state if it satisfies $\mathrm{tr}(\rho) = 1$. In this language, a pure state $|\psi\rangle$ corresponds to the rank-1 Hermitian matrix $|\psi\rangle\langle\psi|$ given by the outer product of the vector with itself.

Linear transformations of an n-qubit quantum system correspond to $2^n \times 2^n$ matrices, called *operators*. Given an operator M, there is always a singular value decomposition (SVD)

$$M = \sum_{i=0}^{2^n-1} \sigma_i |w_i\rangle\langle v_i|\,,$$

where the *singular values* σ_i are non-negative real numbers, and each of the sets $\{|w_i\rangle\}$ and $\{|v_i\rangle\}$ are orthonormal bases for the vector space. If M is Hermitian, then we may call M an *observable*, and in this case, M is guaranteed to have an eigenvalue decomposition

$$M = \sum_{i=0}^{2^n-1} \lambda_i |\psi_i\rangle\langle\psi_i|, \qquad M|\psi_i\rangle = \lambda_i |\psi_i\rangle \;\forall i$$

for which the eigenvalues $\lambda_0, \lambda_1, \ldots, \lambda_{2^n-1}$ are real, and the eigenvectors (also known as *eigenstates*) $|\psi_0\rangle, |\psi_1\rangle, \ldots, |\psi_{2^n-1}\rangle$ form an orthonormal set.

Many end-to-end problems solved by quantum algorithms boil down to estimating the expectation value of an observable, which correspond to a physical property of the system. Given an observable M and a mixed state ρ, the expectation value of M is given by $\mathrm{tr}(M\rho)$. For a pure state $\rho = |\psi\rangle\langle\psi|$, this reduces to $\langle\psi|M|\psi\rangle$.

An important observable is the *Hamiltonian*, which corresponds to the energy of the physical system. The Hamiltonian generates time evolution of the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

state; that is, denoting the state at time t by $|\psi(t)\rangle$ and the time-dependent Hamiltonian by $H(t)$, the state obeys the time-dependent Schrödinger equation

$$\mathrm{i}\frac{\mathrm{d}|\psi(t)\rangle}{\mathrm{d}t} = H(t)|\psi(t)\rangle\,.$$

Here we have set the physical constant $\hbar$ to 1, which is the typical convention in the literature that we cite. The specification of an initial state $|\psi(0)\rangle$ uniquely determines $|\psi(t)\rangle$ for all other t. If $H(t) = H$ is independent of t, this is given exactly by the matrix exponential

$$|\psi(t)\rangle = \mathrm{e}^{-\mathrm{i}Ht}|\psi(0)\rangle\,, \tag{A.1}$$

and if $H(t)$ is time dependent, it is given by a time-ordered matrix exponential.

The eigenvalues of the Hamiltonian are often called the *energies*. The eigenstate corresponding to the minimal eigenvalue is called the *ground state*, and its eigenvalue is called the *ground state energy*; the eigenstates corresponding to larger energies are called *excited states*. In actual quantum systems like atomic nuclei, molecules, and materials, the system's lower energies—and especially its ground state and ground state energy—often determine its key properties; the higher excited states are rarely populated due to energy exchange with the environment favoring lower energy levels. As such, many of the relevant end-to-end problems for which quantum computing may be helpful relate to computing ground state energies and other properties of low-energy states.

One complication is that actual quantum systems in nature typically do not directly correspond to a collection of two-level qubit systems. Instead, they are modeled as discrete or continuous systems with a larger (possibly infinite) number of levels. However, the states of these systems are still described by vectors in a well-defined vector space. For example, the position of an electron in 3D space is given by an element of the vector space of square integrable functions on $\mathbb{R}^3$—in this context, the state vector $|\psi\rangle$ is often called the *wavefunction*, a term that is sometimes also used in discrete situations as well. The position of η particles in 3D space has 3η continuous degrees of freedom, and states correspond to square integrable functions on $\mathbb{R}^{3\eta}$. However, particles found in nature, such as electrons, are indistinguishable, and quantum mechanics dictates that the corresponding wavefunctions must either be antisymmetric (if the particles are fermions) or symmetric (if they are bosons) under particle exchange, which restricts the accessible vector space. For generic multiqubit systems, no such symmetry is naturally imposed. Fermionic and bosonic systems are the subjects of quantum algorithms for chemistry, condensed matter physics, and nuclear and particle physics—to simulate these and other non-qubit systems on a quantum computer, algorithmic choices must be made on

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

how to embed the relevant vector space into a tensor product of qubit systems. It may also be required to truncate the (possibly infinite-dimensional) vector space, incurring errors in the calculation.

A.2 The quantum circuit model

We follow standard convention and work in the quantum circuit model of quantum computation. In this paradigm, quantum computations process the information in quantum states by applying a sequence of unitary operators to the state, known as *gates*, which generalize classical Boolean logic gates—unitarity ensures that the norm of the state is preserved by the operator being applied. A single-qubit gate is given by a 2×2 unitary matrix. There are a few essential examples that occur throughout the book.

- The Pauli matrices (which are both unitary and Hermitian):

$$\sigma_x = X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_y = Y = \begin{pmatrix} 0 & -\mathrm{i} \\ \mathrm{i} & 0 \end{pmatrix}, \qquad \sigma_z = Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix},$$

- The Hadamard gate H and the phase gate S:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \qquad S = \sqrt{Z} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix},$$

- The T gate:

$$T = \sqrt{S} = \begin{pmatrix} 1 & 0 \\ 0 & \mathrm{e}^{\mathrm{i}\pi/4} \end{pmatrix}.$$

A k-qubit gate is given by a $2^k \times 2^k$ unitary matrix, and some of the essential multiqubit gates include the 2-qubit controlled NOT (CNOT) gate and the 3-qubit Toffoli gate

$$\mathrm{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \qquad \mathrm{TOFFOLI} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}.$$

Given as input a 2-qubit state $|x_0 x_1\rangle$ with $x_0, x_1 \in \{0, 1\}$, the CNOT gate flips the value of x_1 conditioned on ("controlled on") x_0 being 1; the first qubit is

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

the control and the second is the target. The Toffoli gate is a doubly controlled NOT gate in the sense that it flips the third qubit controlled on both the first and second qubits being set to 1.

When a k-qubit gate described by the $2^k \times 2^k$ matrix V acts on a subset of qubits in an n-qubit system (with $n > k$), the gate enacted on the n-qubit system is given by a tensor product of V with the identity matrix. For example, if a single-qubit X gate acts on the second qubit of an n-qubit system, then the full $2^n \times 2^n$ unitary operator U for the gate may be decomposed as

$$U = I \otimes X \otimes I \otimes I \otimes \cdots \otimes I\,,$$

where I is the 2×2 identity matrix.

Given a fixed discrete (i.e., finite) set of gates, we may consider the set of all gates generated by the discrete gate set—that is, gates that can be formed by multiplying a sequence of gates drawn from the discrete gate set. A gate set is *universal* if it generates a dense subset of the set of all n-qubit unitary operators on the system, or equivalently, if any n-qubit unitary may be approximated to arbitrary precision by a product of gates drawn from the generating set.

For an n-qubit system, the discrete gate set formed from single-qubit gates $\{X, Y, Z, H, S\}$ on each of the n qubits combined with 2-qubit CNOT gates between any pair of qubits generates the *Clifford* group, which is a finite group that is not universal. Importantly, quantum computations on n qubits involving only Clifford gates can be efficiently simulated on a classical computer; thus, significant quantum computational speedups cannot be achieved using only Clifford gates. By adding either the T gate or the Toffoli gate to the generating set, the gate set becomes universal. The Clifford + T gate set is the most common discrete gate set considered in compilations of quantum algorithms. The Toffoli gate can be exactly decomposed into Clifford gates and T gates.

In this language, a quantum computation consists of the initialization of a quantum state (typically an n-qubit product state such as $|0\rangle^{\otimes n}$), the application of a prespecified sequence of gates, and finally, a measurement of the n qubits in the computational basis. If the normalized n-qubit initial state is $|\psi_0\rangle$ and the sequence of gates is $U_1, \ldots, U_\ell$, then the quantum state prior to the measurement is given by $|\phi\rangle = U_\ell U_{\ell-1} \cdots U_1 |\psi_0\rangle$. The measurement then produces an outcome x with probability equal to $|\langle x|\phi\rangle|^2$. This procedure can be depicted in a *quantum circuit* diagram, such as Fig. A.1.

When performing a *resource estimate* of a quantum algorithm in the Clifford + T gate set, the key quantities are the total number of qubits and gates that appear in the associated quantum circuit diagram. Occasionally, one is also interested in the circuit *depth*. For example, the circuit in Fig. A.1 acts on 3 qubits and has a total gate count equal to 12. The circuit depth is 6 since the

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

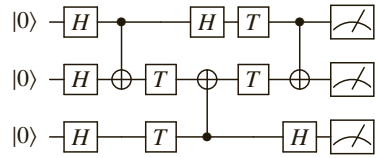


Figure A.1 Example of a quantum circuit with gates drawn from the Clifford + T gate set. Time flows from left to right. The two qubit gates are CNOT gates with the control indicated by the symbol • and the target indicated by the symbol ⊕.

gates can be *parallelized* into 6 sequential layers. When working in the Clifford + T gate set, it is common to ignore the Clifford gates and only count the non-Clifford gates, that is, the T gates (or the Toffoli gates). The main reason for this is that non-Clifford gates are more difficult to implement than Clifford gates in many (but not all) schemes for fault-tolerant quantum computing. The circuit in Fig. A.1 has a T-count of 4 and a T-depth of 2, since only 2 of the 6 layers contain T gates.

A quantum algorithm for a certain computational problem is a procedure that takes as input an instance of the computational problem and determines a quantum circuit (or multiple quantum circuits), as well as a procedure to convert the measurement result(s) into the answer to the computational problem. Since measurement outcomes are random, the answer need only be correct with high probability.

A.3 Noise in quantum gates and the NISQ era

The quantum circuit model is an idealized and abstract depiction of a quantum computer. Actual quantum computers attempt to realize this model by using physical 2-level quantum systems as qubits—several options can be considered including the electronic states of ions or neutral atoms, the spin states of electrons, the polarization states of photons, and the number of excitations in superconducting electrical circuits. Gates are applied by turning on and off external fields. While experimental control of these quantum systems has improved dramatically over time, one cannot expect gates to be performed perfectly. Much of the work in theoretical quantum information science deals with how to characterize, detect, and correct the errors that occur in noisy quantum computations.

Specifically, methods for fault-tolerant quantum computation have been developed, whereby a quantum computation can be accomplished correctly using

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

faulty components, provided that the noise in the system meets certain conditions. The ideal quantum circuit is referred to as the *logical* circuit, composed of logical gates and logical qubits. Each logical qubit is realized using a larger number of noisy *physical qubits*, and each logical gate requires the action of multiple faulty *physical gates*. A resource estimate for the ideal logical circuit can be converted into a resource estimate for the actual physical quantum computer, a calculation that depends on the specifics of the hardware and the fault-tolerance scheme being utilized.

Due to the resource overhead required by fault-tolerant quantum computation, researchers have also investigated the question of whether noisy quantum computers can solve interesting problems without correcting the errors. The era of quantum computing where quantum devices of tens or hundreds of qubits exist, but large-scale fault-tolerant quantum computation is not yet possible, has been referred to as the noisy intermediate-scale quantum (NISQ) era. Generally speaking, algorithms for NISQ-era quantum computers should possess a certain resilience to the inevitable occurrence of errors in the computation, often by restricting to quantum circuits with a limited gate count or gate depth. While the focus of this book is on quantum algorithms for fault-tolerant quantum computers and logical resource estimates, we comment in passing on NISQ algorithms for many of the tasks.

A.4 Big-O notation

Analyses of (classical or quantum) algorithms often focus on how the computational cost, also referred to as the *complexity*, scales with the size of the input. Inputs to a computational problem are assigned an integer size n—for example, the number of digits in a number one wishes to factor—and resource metrics such as the qubit count, gate count, and circuit depth are expressed as functions of n. Often, of primary interest is the asymptotic scaling of these complexities with n. To facilitate this, we adopt big-O notation. Given two positive-valued real functions $f(n)$ and $g(n)$, we use the following definitions:

- $f(n) = O(g(n))$ if there exist n_0, c, such that $f(n) \leq cg(n)$ whenever $n \geq n_0$.
- $f(n) = \Omega(g(n))$ if there exist n_0, c, such that $f(n) \geq cg(n)$ whenever $n \geq n_0$.
- $f(n) = \Theta(g(n))$ if $f(n) = \Omega(g(n))$ and $f(n) = O(g(n))$.
- $f(n) = \widetilde{O}(g(n))$ if there exists c such that $f(n) = O(g(n) \cdot \log^c(g(n)))$.
- $f(n) = \text{poly}(n)$ if there exists c such that $f(n) = O(n^c)$.
- $f(n) = \text{polylog}(n)$ if there exists c such that $f(n) = O(\log(n)^c)$.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Above, n_0 and c are always constants, independent of n. Intuitively, O, Ω, and Θ are used to indicate that the asymptotic growth rate of $f(n)$ is upper bounded, lower bounded, and exactly equal to that of $g(n)$, respectively. Tildes are added to suppress logarithmic factors and simplify the expressions.

We also occasionally utilize little-o notation, which has the following definitions:

- $f(n) = o(g(n))$ if *for any* constant c there exists n_0 for which $f(n) \leq cg(n)$ whenever $n \geq n_0$.
- $f(n) = \omega(g(n))$ if *for any* constant c there exists n_0 for which $f(n) \geq cg(n)$ whenever $n \geq n_0$.

Thus, little-o and little-ω communicate instances where the growth rate of $f(n)$ is *strictly* smaller than $g(n)$ and larger than $g(n)$, respectively.

While big-O and little-o notation carries the formal mathematical definitions above, in some contexts this notation is utilized in a less mathematically precise fashion. For example, big-O is occasionally used simply to indicate that constant prefactors have been omitted or that a certain quantity is roughly of the same order as another. The expression $O(1)$ is often used as a placeholder for an unspecified constant, even when there is not a well-defined growing parameter n. Meanwhile, the expression $o(1)$ is used for functions $f(n)$ that approach 0 as $n \to \infty$. The usage of Ω is often chosen to add emphasis to the fact that a certain quantity is a *lower bound* for another, even when Θ would also have been mathematically appropriate.

We also employ big-O notation for functions of multiple independent parameters. For example, if the input is an $m \times n$ matrix, we might be interested in the complexity dependence on both m and n. Another common scaling parameter is the target precision ϵ to which a certain quantity should be estimated by the quantum algorithm. Smaller ϵ typically incurs greater resources, and thus we wish to compute how the complexity scales with growing $1/\epsilon$. When two multivariate functions f and g are monotonically nondecreasing in all of the scaling parameters, there is little ambiguity about how to extend the definitions. For example:

- $f(n, m) = O(g(n, m))$ if there exist n_0, c, such that $f(n, m) \leq cg(n, m)$ whenever $n, m \geq n_0$.
- $f(n, m) = \text{poly}(n, m)$ if there exists c such that $f(n, m) = O((nm)^c)$.

Ambiguity may arise if it is possible for the function to decrease when certain scaling parameters are increased; in this case, the limiting behavior of the function can depend on the rates at which the parameters grow relative to one

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

another. Additional context about the range or relationship of the scaling parameters may be required to understand what is being communicated by the big-O notation.

Big-O notation enables a determination of the magnitude of a quantum speedup. Suppose the complexity of the quantum algorithm is $Q(n)$ and the complexity of the classical algorithm is $C(n)$.

- We say that the quantum algorithm has an *exponential speedup* if $Q(n) = O(\log(C(n)))$.
- We say that the quantum algorithm has a *polynomial speedup* of degree d if $Q(n) = O(C(n)^{1/d})$. If $Q(n) = \widetilde{O}(C(n)^{1/d})$, we say that the speedup is *essentially* (or *nearly*) degree-d, and often we drop these qualifiers for ease of discussion.
- If $Q(n)$ and $C(n)$ meet the criterion for a polynomial speedup for all $d \geq 1$ but not the criterion for an exponential speedup, then we say the speedup is *superpolynomial*.

Here are some examples:

- If $Q(n) = 3n$ and $C(n) = 2n^3$, there is a degree-3, or *cubic*, polynomial speedup.
- If $Q(n) = n2^{n/4}$ and $C(n) = 2^n$, there is a nearly degree-4, or *quartic*, polynomial speedup.
- If $Q(n) = n^2$ and $C(n) = \mathrm{e}^{n^{1/3}}$, then there is a superpolynomial speedup.
- If $Q(n) = 10n$ and $C(n) = 2^{n/1000}$, then there is an exponential speedup.

End-to-end analyses should ideally also assess the constant prefactors that are omitted when using big-O notation (and polylogarithmic prefactors when using big-$\widetilde{O}$), as these can still contribute significantly to the outlook of a certain application if they are especially large.

A.5 Complexity theory background

Occasionally, we make reference to concepts and results from complexity theory. Complexity theory aims to classify different computational problems based on the quantity of computational resources required to solve them. These computational complexities are often categorized solely based on whether they scale polynomially or superpolynomially with the size of the input. If the complexity of an algorithm is polynomial in the relevant parameter, it is called *efficient*.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Let x be an instance of a computational problem, which is associated with an integer length n. The desired output of the problem on input x is denoted by $M(x)$. If $M(x) \in \{0, 1\}$ is a single bit, the problem is called a *decision problem*. This enables a definition of the following important complexity classes:

- The set P contains decision problems where $M(x)$ can be computed by a deterministic classical algorithm with time complexity poly(n).
- The set BPP contains decision problems where $M(x)$ can be computed with high probability by a randomized classical algorithm (i.e., an algorithm that can make coin flips) with time complexity poly(n).
- The set BQP contains decision problems where $M(x)$ can be computed with high probability by a quantum algorithm with gate complexity poly(n).

To arrive at a precise form for the time or gate complexity, one must specify a particular computational model; in the case of the quantum circuit model, one also needs to specify a gate set, such as Clifford + T. However, for the purpose of these complexity classes, these details are generally unimportant, as they do not change which problems are in P and BQP.

Complexity theory also defines classes of problems for which the solutions are efficient to verify, even if they are not efficient to compute. Specifically, we may fix a verification algorithm with time complexity poly(n) that computes a function $M'(x, y)$ of two inputs, where x has size n and y has size poly(n). We say that y is a witness for x if $M'(x, y) = 1$. The verification algorithm can also be a quantum algorithm, in which case y can be a quantum state $|y\rangle$, which acts as the initial state for a quantum circuit in the quantum circuit model. A state $|y\rangle$ is a witness for x if the quantum verification algorithm produces output $M'(x, |y\rangle) = 1$ with high probability.

- The set NP contains decision problems for which there exists a deterministic classical verification algorithm where, on input x, there exists a witness y if and only if $M(x) = 1$.
- The set QMA contains decision problems for which there exists a quantum verification algorithm where, on input x, there exists a witness $|y\rangle$ if and only if $M(x) = 1$.

The most famous outstanding open question in complexity theory is whether P = NP, that is, whether or not there exist problems that cannot be solved efficiently, but for which solutions can be verified efficiently given a witness. It is widely believed that P ≠ NP. The prototypical example of a problem in NP that is believed not to be in P is the Boolean satisfiability problem. Here, the input is a Boolean formula φ (referred to as x above, and specified by a description of length n). The formula φ maps an input string z consisting of m

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

bits to an output bit $\varphi(z) \in \{0, 1\}$, where $n = \text{poly}(m)$. Given z, the output bit $\varphi(z)$ can be evaluated in time complexity $\text{poly}(n)$, and the question is whether there exists a z for which $\varphi(z) = 1$, in which case we call φ satisfiable. This problem is efficient to verify since whenever $\varphi(z) = 1$, the string z acts as a witness to the fact that φ is satisfiable. However, there are 2^m possible inputs z, so without a witness, it naively requires trying all possible inputs to determine if φ is satisfiable, a procedure which has superpolynomial-in-n time complexity.

In fact, it has been shown that the Boolean satisfiability problem is NP-hard, a term that means it is as hard as any other problem in NP. Specifically, we can make the following (slightly informal) definitions:

- A problem is NP-hard if the existence of an efficient deterministic classical algorithm for the problem would imply that P = NP.
- A problem is QMA-hard if the existence of an efficient quantum algorithm for the problem would imply that BQP = QMA.
- A problem is BQP-hard if the existence of an efficient randomized classical algorithm for the problem would imply that BPP = BQP.[1]

A problem that is both NP-hard and in NP is called NP-complete. The Boolean satisfiability problem is one example of an NP-complete problem. Similar definitions follow for the terms QMA-complete and BQP-complete.

The conjecture that $\text{P} \neq \text{NP}$ entails that all NP-hard problems do not admit efficient classical algorithms. Thus, one way to give evidence that a problem cannot be solved in polynomial time is to prove that it is NP-hard. Since it is also widely believed that $\text{BQP} \neq \text{QMA}$ and that $\text{NP} \not\subset \text{BQP}$, showing that a problem is NP-hard or QMA-hard is strong evidence that it does not admit an efficient quantum algorithm. In the search for good quantum algorithms, these hardness results establish limits on what we expect to be possible.

On the other hand, if one can show that a problem is BQP-complete, then this is evidence that the problem exhibits a superpolynomial quantum speedup over the best possible classical algorithm. If it did not, then this would imply that BPP = BQP, which is widely believed to be false.

These complexity-theoretic results are useful guides for navigating quantum algorithms, but it is worth emphasizing that they typically deal with worst-case hardness and may not always be relevant for real-world instances of a problem. For example, preparing the ground state of a Hamiltonian consisting of local terms—which is in a sense a quantum analog of the Boolean satisfiability problem—is well known to be QMA-complete; thus, it is not expected to admit an efficient quantum algorithm in all instances. Nevertheless, for many specific

[1] Technically, due to their probabilistic nature, BPP and BQP should be defined as classes of *promise* problems to correctly formalize the notion of BQP-hard; see [418].

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Hamiltonians that arise in nature, we do expect efficient ground state preparation to be possible, and this forms the basis for many proposed applications of quantum computing.

We conclude with a discussion of the concept of *oracles*. A (classical or quantum) algorithm is said to have access to an oracle g if it can *query* the oracle in a black-box fashion, by fixing an n-bit input string z and receiving the corresponding m-bit output string $g(z)$. The *query complexity* of an algorithm is the number of times it requests an output from the oracle. In quantum algorithms, one typically allows the oracle to be queried in superposition, in the sense of performing the unitary map U_g, defined by

$$U_g\left(\sum_{z=0}^{2^n-1}\sum_{w=0}^{2^m-1}\alpha_{z,w}|z\rangle|w\rangle\right)=\sum_{z=0}^{2^n-1}\sum_{w=0}^{2^m-1}\alpha_{z,w}|z\rangle|w\oplus g(z)\rangle\,,$$

where $\oplus$ denotes bitwise addition modulo 2, and the coefficients $\alpha_{z,w}$ are arbitrary complex numbers. The ability to query in superposition may give a quantum algorithm an advantage over classical algorithms that cannot do so.

Oracles play multiple conceptual roles. For one, they enable modular accounting of the costs of an algorithm. For example, in algorithms where oracle calls represent the dominant computational burden, one may count the number of oracle queries made by the algorithm and multiply this by the computational cost (e.g., time complexity or gate complexity) required to implement a single oracle query. In end-to-end analyses, it is important to instantiate all oracles using elementary operations and account for the costs of implementing them in final resource expressions.

In complexity theory, oracles also provide a mechanism for establishing more definitive separations between different models of computation. Oracles are black-box objects, and algorithms may only interact with oracles by querying them on different inputs—this makes it easier to prove a lower bound on the query complexity of an algorithm than to prove a lower bound on the time complexity or gate complexity. For example, there exists an oracle g for which it can be shown that $\mathrm{P} \neq \mathrm{NP}$ *relative to* g, even though $\mathrm{P} = \mathrm{NP}$ remains possible in the non-oracle setting (in fact, there also exist oracles relative to which $\mathrm{P} = \mathrm{NP}$).

Similarly, there are specific computational problems involving oracles, such as Simon's problem, where one can show an unconditional exponential separation between the quantum and classical query complexity required to solve the problem. Relative to this oracle, it holds that $\mathrm{BPP} \neq \mathrm{BQP}$. Such separations do not alone constitute a definitive quantum advantage in end-to-end complex-

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ity, but they may capture the core mechanism by which an end-to-end analysis aims to achieve an advantage.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

References

[1] Aaronson, S. (2015). Read the fine print. *Nat. Phys.*, **11**, 291–293. doi:10.1038/nphys3272.

[2] Aaronson, S. (2018). Shadow tomography of quantum states. Pages 325–338 of: *STOC*. doi:10.1145/3188745.3188802. arXiv:1711.01053.

[3] Aaronson, S. (2021). Open problems related to quantum query complexity. *ACM Trans. Quantum Comput.*, **2**. doi:10.1145/3488559. arXiv:2109.06917.

[4] Aaronson, S., and Gottesman, D. (2004). Improved simulation of stabilizer circuits. *Phys. Rev. A*, **70**, 052328. doi:10.1103/PhysRevA.70.052328. arXiv:quant-ph/0406196.

[5] Aaronson, S., Chen, X., Hazan, E., et al. (2019). Online learning of quantum states. *J. Stat. Mech. Theory Exp.*, **2019**, 124019. doi:10.1088/1742-5468/ab3988. arXiv:1802.09025.

[6] Abbas, A., Ambainis, A., Augustino, B., et al. (2024). Challenges and opportunities in quantum optimization. *Nat. Rev. Phys.*, **6**, 718–735. doi:10.1038/s42254-024-00770-9. arXiv:2312.02279.

[7] Adachi, S. H., and Henderson, M. P. (2015). Application of quantum annealing to training of deep neural networks. arXiv:1510.06356.

[8] Aggarwal, D., Brennen, G., Lee, T., et al. (2018). Quantum attacks on bitcoin, and how to protect against them. *Ledger*, **3**. doi:10.5195/ledger.2018.127. arXiv:1710.10377.

[9] Agrawal, A. A., Wilson, T. L., Saadatmand, S., et al. (2024). Quantifying fault tolerant simulation of strongly correlated systems using the Fermi–Hubbard model. arXiv:2406.06511.

[10] Aharonov, D., and Ben-Or, M. (2008). Fault-tolerant quantum computation with constant error rate. *SIAM J. Comp.*, **38**, 1207–1282. doi:10.1137/S0097539799359385. Earlier version in *STOC'97*. arXiv:quant-ph/9906129.

[11] Aharonov, D., and Ta-Shma, A. (2007). Adiabatic quantum state generation. *SIAM J. Comp.*, **37**, 47–82. doi:10.1137/060648829. Earlier version in *STOC'03*. arXiv:quant-ph/0301023.

[12] Aharonov, D., Ben-Or, M., Impagliazzo, R., and Nisan, N. (1996). Limitations of noisy reversible computation. arXiv:quant-ph/9611028.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[13] Akhalwaya, I. Y., Ubaru, S., Clarkson, K. L., et al. (2022). Exponential advantage on noisy quantum computers. arXiv:2209.09371.
[14] Akhalwaya, I. Y., Connolly, A., Guichard, R., et al. (2023). A modular engine for quantum Monte Carlo integration. arXiv:2308.06081.
[15] Alagic, G., Apon, D., Cooper, D., et al. (2022). *Status report on the third round of the NIST post-quantum cryptography standardization process*. Tech. rept. NISTIR 8413. National Institute of Standards and Technology, Gaithersburg, MD. doi:10.6028/NIST.IR.8413-upd1.
[16] Albash, T., and Lidar, D. A. (2018). Adiabatic quantum computation. *Rev. Mod. Phys.*, **90**, 015002. doi:10.1103/RevModPhys.90.015002. arXiv:1611.04471.
[17] Albert, V. V., and Faist, P. (eds). (2024). *The error correction zoo*. https://errorcorrectionzoo.org/, accessed: 2024-12-01.
[18] Alexis, M., Lin, L., Mnatsakanyan, G., et al. (2024). Infinite quantum signal processing for arbitrary Szegő functions. arXiv:2407.05634.
[19] Alexis, M., Mnatsakanyan, G., and Thiele, C. (2024). Quantum signal processing and nonlinear Fourier analysis. *Rev. Mat. Complut.*, **37**, 655–694. doi:10.1007/s13163-024-00494-5. arXiv:2310.12683.
[20] Aliferis, P., Gottesman, D., and Preskill, J. (2006). Quantum accuracy threshold for concatenated distance-3 codes. *Quantum Inf. Comput.*, **6**, 97–165. doi:10.26421/QIC6.2-1. arXiv:quant-ph/0504218.
[21] Aliferis, P., Gottesman, D., and Preskill, J. (2008). Accuracy threshold for postselected quantum computation. *Quantum Inf. Comput.*, **8**, 181–244. doi:10.26421/QIC8.3-4-1. arXiv:quant-ph/0703264.
[22] Altaisky, M. (2001). Quantum neural network. arXiv:quant-ph/0107012.
[23] Altshuler, B., Krovi, H., and Roland, J. (2010). Anderson localization makes adiabatic quantum optimization fail. *Proc. Natl. Acad. Sci.*, **107**, 12446–12450. doi:10.1073/pnas.1002116107. arXiv:0912.0746.
[24] Ambainis, A. (2004). Quantum search algorithms. *SIGACT News*, **35**, 22–35. doi:10.1145/992287.992296. arXiv:quant-ph/0504012.
[25] Ambainis, A. (2007). Quantum walk algorithm for element distinctness. *SIAM J. Comp.*, **37**, 210–239. doi:10.1137/S0097539705447311. Earlier version in *FOCS'04*. arXiv:quant-ph/0311001.
[26] Ambainis, A. (2012). Variable time amplitude amplification and quantum algorithms for linear algebra problems. Pages 636–647 of: *STACS*. doi:10.4230/LIPIcs.STACS.2012.636. arXiv:1010.4458.
[27] Ambainis, A., and Kokainis, M. (2017). Quantum algorithm for tree size estimation, with applications to backtracking and 2-player games. Pages 989–1002 of: *STOC*. doi:10.1145/3055399.3055444. arXiv:1704.06774.
[28] Ambainis, A., Balodis, K., Iraids, J., et al. (2019). Quantum speedups for exponential-time dynamic programming algorithms. Pages 1783–1793 of: *SODA*. doi:10.1137/1.9781611975482.107. arXiv:2104.14384.
[29] Ambainis, A., Kokainis, M., and Vihrovs, J. (2023). Improved algorithm and lower bound for variable time quantum search. arXiv:2302.06749.
[30] Amin, M. H., Andriyash, E., Rolfe, J., et al. (2018). Quantum Boltzmann machine. *Phys. Rev. X*, **8**, 021050. doi:10.1103/PhysRevX.8.021050. arXiv:1601.02036.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[31] An, D., and Lin, L. (2022). Quantum linear system solver based on time-optimal adiabatic quantum computing and quantum approximate optimization algorithm. *ACM Trans. Quantum Comput.*, **3**. doi:10.1145/3498331. arXiv:1909.05500.

[32] An, D., and Trivisa, K. (2023). Quantum algorithms for linear and non-linear fractional reaction–diffusion equations. arXiv:2310.18900.

[33] An, D., Linden, N., Liu, J.-P., et al. (2021). Quantum-accelerated multilevel Monte Carlo methods for stochastic differential equations in mathematical finance. *Quantum*, **5**, 481. doi:10.22331/q-2021-06-24-481. arXiv:2012.06283.

[34] An, D., Fang, D., and Lin, L. (2021). Time-dependent unbounded Hamiltonian simulation with vector norm scaling. *Quantum*, **5**, 459. doi:10.22331/q-2021-05-26-459. arXiv:2012.13105.

[35] An, D., Liu, J.-P., Wang, D., and Zhao, Q. (2022). A theory of quantum differential equation solvers: Limitations and fast-forwarding. arXiv:2211.05246.

[36] An, D., Liu, J.-P., and Lin, L. (2023). Linear combination of Hamiltonian simulation for nonunitary dynamics with optimal state preparation cost. *Phys. Rev. Lett.*, **131**, 150603. doi:10.1103/PhysRevLett.131.150603. arXiv:2303.01029.

[37] An, D., Childs, A. M., and Lin, L. (2023). Quantum algorithm for linear non-unitary dynamics with near-optimal dependence on all parameters. arXiv:2312.03916.

[38] Andersen, E. D., and Andersen, K. D. (2000). The Mosek interior point optimizer for linear programming: An implementation of the homogeneous algorithm. Pages 197–232 of: *High Performance Optimization*. doi:10.1007/978-1-4757-3216-0_8.

[39] Anschuetz, E., Olson, J., Aspuru-Guzik, A., and Cao, Y. (2019). Variational quantum factoring. Pages 74–85 of: *Quantum Technology and Optimization Problems*. doi:10.1007/978-3-030-14082-3_7. arXiv:1808.08927.

[40] Anschuetz, E. R., and Cao, Y. (2019). Realizing quantum Boltzmann machines through eigenstate thermalization. arXiv:1903.01359.

[41] Anschuetz, E. R., and Kiani, B. T. (2022). Quantum variational algorithms are swamped with traps. *Nat. Commun.*, **13**, 7760. doi:10.1038/s41467-022-35364-5. arXiv:2205.05786.

[42] Anwar, H., Brown, B. J., Campbell, E. T., and Browne, D. E. (2014). Fast decoders for qudit topological codes. *New J. Phys.*, **16**, 063038. doi:10.1088/1367-2630/16/6/063038. arXiv:1311.4895.

[43] Aoki, H., Tsuji, N., Eckstein, M., et al. (2014). Nonequilibrium dynamical mean-field theory and its applications. *Rev. Mod. Phys.*, **86**, 779–837. doi:10.1103/RevModPhys.86.779. arXiv:1310.5329.

[44] Aoki, Y., Blum, T., Colangelo, G., et al. (2022). FLAG review 2021. *Eur. Phys. J. C*, **82**, 869. arXiv:2111.09849.

[45] van Apeldoorn, J., and Gilyén, A. (2019). Improvements in quantum SDP-solving with applications. Pages 99:1–99:15 of: *ICALP*. doi:10.4230/LIPIcs.ICALP.2019.99. arXiv:1804.05058.

[46] van Apeldoorn, J., and Gilyén, A. (2019). Quantum algorithms for zero-sum games. arXiv:1904.03180.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[47] van Apeldoorn, J., Gilyén, A., Gribling, S., and de Wolf, R. (2020). Convex optimization using quantum oracles. *Quantum*, **4**, 220. doi:10.22331/q-2020-01-13-220. arXiv:1809.00643.

[48] van Apeldoorn, J., Gilyén, A., Gribling, S., and de Wolf, R. (2020). Quantum SDP-solvers: Better upper and lower bounds. *Quantum*, **4**, 230. doi:10.22331/q-2020-02-14-230. Earlier version in *FOCS'17*. arXiv:1705.01843.

[49] van Apeldoorn, J., Cornelissen, A., Gilyén, A., and Nannicini, G. (2023). Quantum tomography using state-preparation unitaries. Pages 1265–1318 of: *SODA*. doi:10.1137/1.9781611977554.ch47. arXiv:2207.08800.

[50] Apers, S., and de Wolf, R. (2020). Quantum speedup for graph sparsification, cut approximation and Laplacian solving. Pages 637–648 of: *FOCS*. doi:10.1109/FOCS46700.2020.00065. arXiv:1911.07306.

[51] Apers, S., and Gribling, S. (2023). Quantum speedups for linear programming via interior point methods. arXiv:2311.03215.

[52] Apers, S., Gilyén, A., and Jeffery, S. (2021). A unified framework of quantum walk search. Pages 6:1–6:13 of: *STACS*. doi:10.4230/LIPIcs.STACS.2021.6. arXiv:1912.04233.

[53] Apers, S., Gribling, S., Sen, S., and Szabó, D. (2023). A (simple) classical algorithm for estimating Betti numbers. *Quantum*, **7**, 1202. doi:10.22331/q-2023-12-06-1202. arXiv:2211.09618.

[54] Arad, I., and Landau, Z. (2010). Quantum computation and the evaluation of tensor networks. *SIAM J. Comp.*, **39**, 3089–3121. doi:10.1137/080739379. arXiv:0805.0040.

[55] Argüello-Luengo, J., González-Tudela, A., Shi, T., et al. (2019). Analogue quantum chemistry simulation. *Nature*, **574**, 215–218. doi:10.1038/s41586-019-1614-4. arXiv:1807.09228.

[56] Arora, S., and Kale, S. (2007). A combinatorial, primal-dual approach to semidefinite programs. Pages 227–236 of: *STOC*. doi:10.1145/1250790.1250823.

[57] Arora, S., Hazan, E., and Kale, S. (2005). Fast algorithms for approximate semidefinite programming using the multiplicative weights update method. Pages 339–348 of: *FOCS*. doi:10.1109/SFCS.2005.35.

[58] Arora, S., Hazan, E., and Kale, S. (2012). The multiplicative weights update method: a meta-algorithm and applications. *Theory Comput.*, **8**, 121–164. doi:10.4086/toc.2012.v008a006.

[59] Arovas, D. P., Berg, E., Kivelson, S. A., and Raghu, S. (2022). The Hubbard model. *Annu. Rev. Condens. Matter Phys.*, **13**, 239–274. doi:10.1146/annurev-conmatphys-031620-102024. arXiv:2103.12097.

[60] Arrasmith, A., Holmes, Z., Cerezo, M., and Coles, P. J. (2022). Equivalence of quantum barren plateaus to cost concentration and narrow gorges. *Quantum Sci. Technol.*, **7**, 045015. doi:10.1088/2058-9565/ac7d06. arXiv:2104.05868.

[61] Arrazola, J. M., Delgado, A., Bardhan, B. R., and Lloyd, S. (2020). Quantum-inspired algorithms in practice. *Quantum*, **4**, 307. doi:10.22331/q-2020-08-13-307. arXiv:1905.10415.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[62] Arunachalam, S., Gheorghiu, V., Jochym-O'Connor, T., et al. (2015). On the robustness of bucket brigade quantum RAM. *New J. Phys.*, **17**, 123010. doi:10.1088/1367-2630/17/12/123010. arXiv:1502.03450.

[63] Arute, F., Arya, K., Babbush, R., et al. (2019). Quantum supremacy using a programmable superconducting processor. *Nature*, **574**, 505–510. doi:10.1038/s41586-019-1666-5.

[64] Arute, F., Arya, K., Babbush, R., et al. (2020). Observation of separated dynamics of charge and spin in the Fermi–Hubbard model. arXiv:2010.07965.

[65] Ashikhmin, A., Lai, C. Y., and Brun, T. A. (2020). Quantum data-syndrome codes. *IEEE J. Sel. Areas Commun.*, **38**, 449–462. doi:10.1109/JSAC.2020.2968997. arXiv:1907.01393.

[66] Aspuru-Guzik, A., Dutoi, A. D., Love, P. J., and Head-Gordon, M. (2005). Simulated quantum computation of molecular energies. *Science*, **309**, 1704–1707. doi:10.1126/science.1113479. arXiv:0604193.

[67] Atas, Y. Y., Zhang, J., Lewis, R., et al. (2021). SU(2) hadrons on a quantum computer via a variational approach. *Nat. Commun.*, **12**, 6499. doi:10.1038/s41467-021-26825-4. arXiv:2102.08920.

[68] Augustino, B., Terlaky, T., and Zuluaga, L. F. (2022). *An inexact-feasible quantum interior point method for second-order cone optimization*. Tech. rept. 21T-009. Department of Industrial and Systems Engineering, Lehigh University.

[69] Augustino, B., Leng, J., Nannicini, G., et al. (2023). A quantum central path algorithm for linear optimization. arXiv:2311.03977.

[70] Augustino, B., Nannicini, G., Terlaky, T., and Zuluaga, L. F. (2023). Quantum interior point methods for semidefinite optimization. *Quantum*, **7**, 1110. doi:10.22331/q-2023-09-11-1110. arXiv:2112.06025.

[71] Augustino, B., Nannicini, G., Terlaky, T., and Zuluaga, L. (2023). Solving the semidefinite relaxation of QUBOs in matrix multiplication time, and faster with a quantum computer. arXiv:2301.04237.

[72] Babbush, R., McClean, J., Wecker, D., et al. (2015). Chemical basis of Trotter–Suzuki errors in quantum chemistry simulation. *Phys. Rev. A*, **91**, 022311. doi:10.1103/PhysRevA.91.022311. arXiv:1410.8159.

[73] Babbush, R., Berry, D. W., Kivlichan, I. D., et al. (2016). Exponentially more precise quantum simulation of fermions in second quantization. *New J. Phys.*, **18**, 033032. doi:10.1088/1367-2630/18/3/033032. arXiv:1506.01020.

[74] Babbush, R., Berry, D. W., Sanders, Y. R., et al. (2017). Exponentially more precise quantum simulation of fermions in the configuration interaction representation. *Quantum Sci. Technol.*, **3**, 015006. doi:10.1088/2058-9565/aa9463. arXiv:1506.01029.

[75] Babbush, R., Gidney, C., Berry, D. W., et al. (2018). Encoding electronic spectra in quantum circuits with linear T complexity. *Phys. Rev. X*, **8**, 041015. doi:10.1103/PhysRevX.8.041015. arXiv:1805.03662.

[76] Babbush, R., Wiebe, N., Mcclean, J., et al. (2018). Low-depth quantum simulation of materials. *Phys. Rev. X*, **8**, 011044. doi:10.1103/PhysRevX.8.011044. arXiv:1706.00023.

[77] Babbush, R., Berry, D. W., McClean, J. R., and Neven, H. (2019). Quantum simulation of chemistry with sublinear scaling in basis size. *npj Quant. Inf.*, **5**, 92. doi:10.1038/s41534-019-0199-y. arXiv:1807.09802.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[78] Babbush, R., Berry, D. W., and Neven, H. (2019). Quantum simulation of the Sachdev–Ye–Kitaev model by asymmetric qubitization. *Phys. Rev. A*, **99**, 040301. doi:10.1103/PhysRevA.99.040301. arXiv:1806.02793.

[79] Babbush, R., McClean, J. R., Newman, M., et al. (2021). Focus beyond quadratic speedups for error-corrected quantum advantage. *PRX Quantum*, **2**, 010103. doi:10.1103/PRXQuantum.2.010103. arXiv:2011.04149.

[80] Babbush, R., Berry, D. W., Kothari, R., et al. (2023). Exponential quantum speedup in simulating coupled classical oscillators. arXiv:2303.13012.

[81] Babbush, R., Huggins, W. J., Berry, D. W., et al. (2023). Quantum simulation of exact electron dynamics can be more efficient than classical mean-field methods. *Nat. Commun.*, **14**, 4058. doi:10.1038/s41467-023-39024-0. arXiv:2301.01203.

[82] Bagherimehrab, M., Sanders, Y. R., Berry, D. W., et al. (2022). Nearly optimal quantum algorithm for generating the ground state of a free quantum field theory. *PRX Quantum*, **3**, 020364. doi:10.1103/PRXQuantum.3.020364. arXiv:2110.05708.

[83] Bagherimehrab, M., Nakaji, K., Wiebe, N., and Aspuru-Guzik, A. (2023). Fast quantum algorithm for differential equations. arXiv:2306.11802.

[84] Baiardi, A., Stein, C. J., Barone, V., and Reiher, M. (2017). Vibrational density matrix renormalization group. *J. Chem. Theory Comput.*, **13**, 3764–3777. doi:10.1021/acs.jctc.7b00329. arXiv:1703.09313.

[85] Baker, J. S., and Radha, S. K. (2022). Wasserstein solution quality and the quantum approximate optimization algorithm: A portfolio optimization case study. arXiv:2202.06782.

[86] Bakshi, A., and Tang, E. (2024). An improved classical singular value transformation for quantum machine learning. Pages 2398–2453 of: *SODA*. doi:10.1137/1.9781611977912.86. arXiv:2303.01492.

[87] Banaszek, K., Cramer, M., and Gross, D. (2013). Focus on quantum tomography. *New J. Phys.*, **15**, 125020. doi:10.1088/1367-2630/15/12/125020.

[88] Banchi, L., Pereira, J., and Pirandola, S. (2021). Generalization in quantum machine learning: A quantum information standpoint. *PRX Quantum*, **2**, 040321. doi:10.1103/PRXQuantum.2.040321. arXiv:2102.08991.

[89] Banegas, G., and Bernstein, D. J. (2018). Low-communication parallel quantum multi-target preimage search. Pages 325–335 of: *SAC*. ePrint:2017/789.

[90] Banuls, M.-C., Pérez-García, D., Wolf, M. M., et al. (2008). Sequentially generated states for the study of two-dimensional systems. *Phys. Rev. A*, **77**, 052306. doi:10.1103/PhysRevA.77.052306. arXiv:0802.2472.

[91] Banuls, M. C., Cichy, K., Cirac, J. I., et al. (2019). Tensor networks and their use for lattice gauge theories. Page 022 of: *LATTICE*. doi:10.22323/1.334.0022. arXiv:1810.12838.

[92] Bañuls, M. C., Blatt, R., Catani, J., et al. (2020). Simulating lattice gauge theories within quantum technologies. *Eur. Phys. J. D*, **74**, 165. doi:10.1140/epjd/e2020-100571-8. arXiv:1911.00003.

[93] Bao, N., Hayden, P., Salton, G., and Thomas, N. (2015). Universal quantum computation by scattering in the Fermi–Hubbard model. *New J. Phys.*, **17**, 093028. doi:10.1088/1367-2630/17/9/093028. arXiv:1409.3585.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[94] Barahona, F. (1982). On the computational complexity of Ising spin glass models. *J. Phys. A*, **15**, 3241–3253. doi:10.1088/0305-4470/15/10/028.

[95] Barker, E. (2020). *Recommendation for key management: Part 1—general*. Tech. rept. SP 800-57 Part 1 Rev. 5. National Institute of Standards and Technology, Gaithersburg, MD. doi:10.6028/NIST.SP.800-57pt1r5.

[96] Barker, E., and Dang, Q. (2015). *Recommendation for key management, part 3: Application-specific key management guidance*. Tech. rept. SP 800-57 Part 3 Rev. 1. National Institute of Standards and Technology, Gaithersburg, MD. doi:10.6028/NIST.SP.800-57pt3r1.

[97] Barmes, I., and Bosch, B. (2019). *Quantum computers and the bitcoin blockchain*. https://deloitte.com/nl/nl/pages/innovatie/artikelen/quantum-computers-and-the-bitcoin-blockchain.html, accessed: 2023-09-30.

[98] Barmes, I., Bosch, B., and Verdonk, M. (2022). *Quantum risk to the Ethereum blockchain—a bump in the road or a brick wall?* https://deloitte.com/nl/nl/pages/risk/articles/quantum-risk-to-the-ethereum-blockchain.html, accessed: 2023-09-30.

[99] Barone, V., Alessandrini, S., Biczysko, M., et al. (2021). Computational molecular spectroscopy. *Nat. Rev. Methods Primers*, **1**, 38. doi:10.1038/s43586-021-00034-1.

[100] Barrett, B. R., Navratil, P., and Vary, J. P. (2013). Ab initio no core shell model. *Prog. Part. Nucl. Phys.*, **69**, 131–181. doi:10.1016/j.ppnp.2012.10.003.

[101] Bärtschi, A., Caravelli, F., Coffrin, C., et al. (2024). Potential applications of quantum computing at Los Alamos National Laboratory. arXiv:2406.06625.

[102] Bauer, B., Bravyi, S., Motta, M., and Chan, G. K.-L. (2020). Quantum algorithms for quantum chemistry and quantum materials science. *Chem. Rev.*, **120**, 12685–12717. doi:10.1021/acs.chemrev.9b00829. arXiv:2001.03685.

[103] Bauer, C. W., Nachman, B., and Freytsis, M. (2021). Simulating collider physics on quantum computers using effective field theories. *Phys. Rev. Lett.*, **127**, 212001. doi:10.1103/PhysRevLett.127.212001. arXiv:2102.05044.

[104] Bauer, C. W., Davoudi, Z., Balantekin, A. B., et al. (2023). Quantum simulation for high-energy physics. *PRX Quantum*, **4**, 027001. doi:10.1103/PRXQuantum.4.027001. arXiv:2204.03381.

[105] Bauer, C. W., Davoudi, Z., Klco, N., and Savage, M. J. (2023). Quantum simulation of fundamental particles and forces. *Nat. Rev. Phys.*, **5**, 420–432. doi:10.1038/s42254-023-00599-8.

[106] Bausch, J. (2022). Fast black-box quantum state preparation. *Quantum*, **6**. doi:10.22331/q-2022-08-04-773. arXiv:2009.10709.

[107] Bausch, J., Senior, A. W., Heras, F. J. H., et al. (2023). Learning to decode the surface code with a recurrent, transformer-based neural network. arXiv:2310.05900.

[108] Beane, S., Detmold, W., Orginos, K., and Savage, M. (2011). Nuclear physics from lattice QCD. *Prog. Part. Nuc. Phys.*, **66**, 1–40. doi:10.1016/j.ppnp.2010.08.002. arXiv:1004.2935.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[109] Beauregard, S. (2003). Circuit for Shor's algorithm using 2n+3 qubits. *Quantum Inf. Comput.*, **3**, 175–185. doi:10.26421/QIC3.2-8. arXiv:quant-ph/0205095.

[110] Beck, D., Carlson, J., Davoudi, Z., et al. (2023). Quantum information science and technology for nuclear physics: Input into U.S. long-range planning, 2023. arXiv:2303.00113.

[111] Beckman, D., Chari, A. N., Devabhaktuni, S., and Preskill, J. (1996). Efficient networks for quantum factoring. *Phys. Rev. A*, **54**, 1034–1063. doi:10.1103/PhysRevA.54.1034. arXiv:quant-ph/9602016.

[112] Beer, K., Bondarenko, D., Farrelly, T., et al. (2020). Training deep quantum neural networks. *Nat. Commun.*, **11**, 808. doi:10.1038/s41467-020-14454-2. arXiv:1902.10445.

[113] Bellonzi, N., Kunitsa, A., Cantin, J. T., et al. (2024). Feasibility of accelerating homogeneous catalyst discovery with fault-tolerant quantum computers. arXiv:2406.06335.

[114] Ben-Or, M., Gottesman, D., and Hassidim, A. (2013). Quantum refrigerator. arXiv:1301.1995.

[115] Bender, M., Bernard, R., Bertsch, G., et al. (2020). Future of nuclear fission theory. *J. Phys. G*, **47**, 113002. doi:10.1088/1361-6471/abab4f. arXiv:2005.10216.

[116] Benedetti, M., Realpe-Gómez, J., Biswas, R., and Perdomo-Ortiz, A. (2017). Quantum-assisted learning of hardware-embedded probabilistic graphical models. *Phys. Rev. X*, **7**, 041052. doi:10.1103/PhysRevX.7.041052. arXiv:1609.02542.

[117] Benedetti, M., Garcia-Pintos, D., Perdomo, O., et al. (2019). A generative modeling approach for benchmarking and training shallow quantum circuits. *npj Quant. Inf.*, **5**, 1–9. doi:10.1038/s41534-019-0157-8. arXiv:1801.07686.

[118] Benedetti, M., Lloyd, E., Sack, S., and Fiorentini, M. (2019). Parameterized quantum circuits as machine learning models. *Quantum Sci. Technol.*, **4**, 043001. doi:10.1088/2058-9565/ab4eb5. arXiv:1906.07682.

[119] Benedetti, M., Fiorentini, M., and Lubasch, M. (2021). Hardware-efficient variational quantum algorithms for time evolution. *Phys. Rev. Res.*, **3**, 033083. doi:10.1103/PhysRevResearch.3.033083. arXiv:2009.12361.

[120] Benioff, P. (1980). The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines. *J. Stat. Phys.*, **22**, 563–591. doi:10.1007/BF01011339.

[121] Bennett, C. H., and Brassard, G. (2014). Quantum cryptography: Public key distribution and coin tossing. *Theor. Comput. Sci.*, **560**, 7–11. doi:10.1016/j.tcs.2014.05.025. arXiv:2003.06557.

[122] Bennett, C. H., Bernstein, E., Brassard, G., and Vazirani, U. (1997). Strengths and weaknesses of quantum computing. *SIAM J. Comp.*, **26**, 1510–1523. doi:10.1137/S0097539796300933. arXiv:quant-ph/9701001.

[123] Bergholm, V., Izaac, J., Schuld, M., et al. (2018). Pennylane: Automatic differentiation of hybrid quantum-classical computations. arXiv:1811.04968. https://pennylane.ai/codebook, accessed: 2024-12-01.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[124] van den Berg, E., and Temme, K. (2020). Circuit optimization of Hamiltonian simulation by simultaneous diagonalization of Pauli clusters. *Quantum*, **4**, 322. doi:10.22331/q-2020-09-12-322. arXiv:2003.13599.

[125] Bermudez, A., Xu, X., Nigmatullin, R., et al. (2017). Assessing the progress of trapped-ion processors towards fault-tolerant quantum computation. *Phys. Rev. X*, **7**, 041061. doi:10.1103/physrevx.7.041061. arXiv:1705.02771.

[126] Bernstein, D. J., and Lange, T. (2017). Post-quantum cryptography. *Nature*, **549**, 188–194. doi:10.1038/nature23461. ePrint:2017/314.

[127] Bernstein, D. J., Engels, S., Lange, T., et al. (2016). Faster elliptic-curve discrete logarithms on FPGAs. ePrint:2016/382.

[128] Bernstein, D. J., Biasse, J.-F., and Mosca, M. (2017). A low-resource quantum factoring algorithm. Pages 330–346 of: *PQCrypto*. doi:10.1007/978-3-319-59879-6_19. ePrint:2017/352.

[129] Bernstein, E., and Vazirani, U. (1997). Quantum complexity theory. *SIAM J. Comp.*, **26**, 1411–1473. doi:10.1137/S0097539796300921. Earlier version in *STOC'93*.

[130] Berntson, B. K., and Sünderhauf, C. (2024). Complementary polynomials in quantum signal processing. arXiv:2406.04246.

[131] Berry, D. W. (2014). High-order quantum algorithm for solving linear differential equations. *J. Phys. A*, **47**, 105301. doi:10.1088/1751-8113/47/10/105301.

[132] Berry, D. W., and C. S. Costa, P. (2024). Quantum algorithm for time-dependent differential equations using Dyson series. *Quantum*, **8**, 1369. doi:10.22331/q-2024-06-13-1369. arXiv:2212.03544.

[133] Berry, D. W., and Childs, A. M. (2012). Black-box Hamiltonian simulation and unitary implementation. *Quantum Inf. Comput.*, **12**, 29–62. doi:10.26421/QIC12.1-2. arXiv:0910.4157.

[134] Berry, D. W., Ahokas, G., Cleve, R., and Sanders, B. C. (2007). Efficient quantum algorithms for simulating sparse Hamiltonians. *Commun. Math. Phys.*, **270**, 359–371. doi:10.1007/s00220-006-0150-x. arXiv:quant-ph/0508139.

[135] Berry, D. W., Childs, A. M., Cleve, R., et al. (2014). Exponential improvement in precision for simulating sparse Hamiltonians. Pages 283–292 of: *STOC*. doi:10.1145/2591796.2591854. arXiv:1312.1414.

[136] Berry, D. W., Childs, A. M., and Kothari, R. (2015). Hamiltonian simulation with nearly optimal dependence on all parameters. Pages 792–809 of: *FOCS*. doi:10.1109/FOCS.2015.54. arXiv:1501.01715.

[137] Berry, D. W., Childs, A. M., Cleve, R., et al. (2015). Simulating Hamiltonian dynamics with a truncated Taylor series. *Phys. Rev. Lett.*, **114**, 090502. doi:10.1103/PhysRevLett.114.090502. arXiv:1412.4687.

[138] Berry, D. W., Childs, A. M., Ostrander, A., and Wang, G. (2017). Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Commun. Math. Phys.*, **356**, 1057–1081. doi:10.1007/s00220-017-3002-y. arXiv:1701.03684.

[139] Berry, D. W., Kieferová, M., Scherer, A., et al. (2018). Improved techniques for preparing eigenstates of fermionic Hamiltonians. *npj Quant. Inf.*, **4**, 22. doi:10.1038/s41534-018-0071-5. arXiv:1711.10460.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[140] Berry, D. W., Gidney, C., Motta, M., et al. (2019). Qubitization of arbitrary basis quantum chemistry leveraging sparsity and low rank factorization. *Quantum*, **3**, 208. doi:10.22331/q-2019-12-02-208. arXiv:1902.02134.

[141] Berry, D. W., Childs, A. M., Su, Y., et al. (2020). Time-dependent Hamiltonian simulation with L^1-norm scaling. *Quantum*, **4**, 254. doi:10.22331/q-2020-04-20-254. arXiv:1906.07115.

[142] Berry, D. W., Rubin, N. C., Elnabawy, A. O., et al. (2023). Quantum simulation of realistic materials in first quantization using non-local pseudopotentials. arXiv:2312.07654.

[143] Berry, D. W., Su, Y., Gyurik, C., et al. (2024). Analyzing prospects for quantum advantage in topological data analysis. *PRX Quantum*, **5**, 010319. doi:10.1103/PRXQuantum.5.010319. arXiv:2209.13581.

[144] Berry, D. W., Tong, Y., Khattar, T., et al. (2024). Rapid initial state preparation for the quantum simulation of strongly correlated molecules. arXiv:2409.11748.

[145] Beullens, W. (2022). Breaking rainbow takes a weekend on a laptop. Pages 464–479 of: *CRYPTO*. doi:10.1007/978-3-031-15979-4_16. ePrint:2022/214.

[146] Beverland, M. E., Kubica, A., and Svore, K. M. (2021). Cost of universality: A comparative study of the overhead of state distillation and code switching with color codes. *PRX Quantum*, **2**, 020341. doi:10.1103/PRXQuantum.2.020341. arXiv:2101.02211.

[147] Beverland, M. E., Murali, P., Troyer, M., et al. (2022). Assessing requirements to scale to practical quantum advantage. arXiv:2211.07629.

[148] Bharti, K., Cervera-Lierta, A., Kyaw, T. H., et al. (2022). Noisy intermediate-scale quantum algorithms. *Rev. Mod. Phys.*, **94**, 015004. doi:10.1103/RevModPhys.94.015004. arXiv:2101.08448.

[149] Bhaskar, M. K., Hadfield, S., Papageorgiou, A., and Petras, I. (2016). Quantum algorithms and circuits for scientific computing. *Quantum Inf. Comput.*, **16**. doi:10.26421/QIC16.3-4-2. arXiv:1511.08253.

[150] Biamonte, J., and Bergholm, V. (2017). Tensor networks in a nutshell. arXiv:1708.00006.

[151] Biamonte, J., Wittek, P., Pancotti, N., et al. (2017). Quantum machine learning. *Nature*, **549**, 195–202. doi:10.1038/nature23474. arXiv:1611.09347.

[152] Bilgin, E., and Boixo, S. (2010). Preparing thermal states of quantum systems by dimension reduction. *Phys. Rev. Lett.*, **105**, 170405. doi:10.1103/PhysRevLett.105.170405. arXiv:1008.4162.

[153] Binder, S., Ekström, A., Hagen, G., et al. (2016). Effective field theory in the harmonic oscillator basis. *Phys. Rev. C*, **93**, 044332. doi:10.1103/PhysRevC.93.044332. arXiv:1512.03802.

[154] Bittel, L., and Kliesch, M. (2021). Training variational quantum algorithms is NP-hard. *Phys. Rev. Lett.*, **127**, 120502. doi:10.1103/PhysRevLett.127.120502. arXiv:2101.07267.

[155] Blais, A., Grimsmo, A. L., Girvin, S. M., and Wallraff, A. (2021). Circuit quantum electrodynamics. *Rev. Mod. Phys.*, **93**, 025005. doi:10.1103/RevModPhys.93.025005. arXiv:2005.12667.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[156] Blanchet, J., Szegedy, M., and Wang, G. (2024). Quadratic speed-up in infinite variance quantum Monte Carlo. arXiv:2401.07497.

[157] Bloch, I., Dalibard, J., and Nascimbène, S. (2012). Quantum simulations with ultracold quantum gases. *Nat. Phys.*, **8**, 267–276. doi:10.1038/nphys2259.

[158] Blunt, N. S., Camps, J., Crawford, O., et al. (2022). Perspective on the current state-of-the-art of quantum computing for drug discovery applications. *J. Chem. Theory Comput.*, **18**, 7001–7023. doi:10.1021/acs.jctc.2c00574. arXiv:2206.00551.

[159] Blunt, N. S., Caune, L., Izsák, R., et al. (2023). Statistical phase estimation and error mitigation on a superconducting quantum processor. *PRX Quantum*, **4**, 040341. doi:10.1103/PRXQuantum.4.040341. arXiv:2304.05126.

[160] Bogdanov, A., Khovratovich, D., and Rechberger, C. (2011). Biclique cryptanalysis of the full AES. Pages 344–371 of: *ASIACRYPT*. doi:10.1007/978-3-642-25385-0_19. ePrint:2011/449.

[161] Boixo, S., and Somma, R. D. (2010). Necessary condition for the quantum adiabatic approximation. *Phys. Rev. A*, **81**, 032308. doi:10.1103/PhysRevA.81.032308. arXiv:0911.1362.

[162] Boixo, S., Knill, E., and Somma, R. D. (2010). Fast quantum algorithms for traversing paths of eigenstates. arXiv:1005.3034.

[163] Bolte, J., Boustany, R., Pauwels, E., and Pesquet-Popescu, B. (2022). Nonsmooth automatic differentiation: A cheap gradient principle and other complexity results. arXiv:2206.01730.

[164] Bolusani, S., Besançon, M., Bestuzheva, K., et al. (2024). The SCIP optimization suite 9.0. arXiv:2402.17702.

[165] Bombín, H. (2015). Gauge color codes: Optimal transversal gates and gauge fixing in topological stabilizer codes. *New J. Phys.*, **17**, 083002. doi:10.1088/1367-2630/17/8/083002. arXiv:1311.0879.

[166] Bombín, H. (2015). Single-shot fault-tolerant quantum error correction. *Phys. Rev. X*, **5**, 031043. doi:10.1103/physrevx.5.031043. arXiv:1404.5504.

[167] Bombín, H. (2018). 2D quantum computation with 3D topological codes. arXiv:1810.09571.

[168] Bombín, H., and Martin-Delgado, M. (2007). Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates. *Phys. Rev. B*, **75**, 075103. doi:10.1103/PhysRevB.75.075103. arXiv:cond-mat/0607736.

[169] Bombín, H., and Martin-Delgado, M. A. (2006). Topological quantum distillation. *Phys. Rev. Lett.*, **97**, 180501. doi:10.1103/PhysRevLett.97.180501. arXiv:quant-ph/0605138.

[170] Bombín, H., Dawson, C., Liu, Y.-H., et al. (2023). Modular decoding: Parallelizable real-time decoding for quantum computers. arXiv:2303.04846.

[171] Bombín, H. G. D.-C., and Poulin, D. (2012). Universal topological phase of two-dimensional stabilizer codes. *New J. Phys.*, **14**, 073048. doi:10.1088/1367-2630/14/7/073048. arXiv:1103.4606.

[172] Bonfà, P., Frassineti, J., Isah, M. M., et al. (2021). UNDI: An open-source library to simulate muon-nuclear interactions in solids. *Comput. Phys. Commun.*, **260**, 107719. doi:10.1016/j.cpc.2020.107719.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[173] Bonilla Ataides, J. P., Tuckett, D. K., Bartlett, S. D., et al. (2021). The XZZX surface code. *Nat. Commun.*, **12**, 2172. doi:10.1038/s41467-021-22274-1. arXiv:2009.07851.

[174] Bosse, J. L., and Montanaro, A. (2022). Probing ground-state properties of the kagome antiferromagnetic Heisenberg model using the variational quantum eigensolver. *Phys. Rev. B*, **105**, 094409. doi:10.1103/PhysRevB.105.094409. arXiv:2108.08086.

[175] Boudot, F., Gaudry, P., Guillevic, A., et al. (2020). Comparing the difficulty of factorization and discrete logarithm: A 240-digit experiment. Pages 62–91 of: *CRYPTO*. doi:10.1007/978-3-030-56880-1_3. arXiv:2006.06197.

[176] Bouland, A., van Dam, W., Joorati, H., et al. (2020). Prospects and challenges of quantum finance. arXiv:2011.06492.

[177] Bouland, A., Dandapani, A., and Prakash, A. (2023). A quantum spectral method for simulating stochastic processes, with applications to Monte Carlo. arXiv:2303.06719.

[178] Bouland, A., Getachew, Y. M., Jin, Y., et al. (2023). Quantum speedups for zero-sum games via improved dynamic Gibbs sampling. Pages 2932–2952 of: *ICML*. arXiv:2301.03763.

[179] Boulebnane, S., and Montanaro, A. (2024). Solving Boolean satisfiability problems with the quantum approximate optimization algorithm. *PRX Quantum*, **5**, 030348. doi:10.1103/PRXQuantum.5.030348. arXiv:2208.06909.

[180] Boyd, S., and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press. doi:10.1017/CBO9780511804441.

[181] Brandão, F. G. S. L., and Svore, K. M. (2017). Quantum speed-ups for solving semidefinite programs. Pages 415–426 of: *FOCS*. doi:10.1109/FOCS.2017.45. arXiv:1609.05537.

[182] Brandão, F. G. S. L., Kalev, A., Li, T., et al. (2019). Quantum SDP solvers: Large speed-ups, optimality, and applications to quantum learning. Pages 27:1–27:14 of: *ICALP*. doi:10.4230/LIPIcs.ICALP.2019.27. arXiv:1710.02581.

[183] Brandão, F. G. S. L., Kueng, R., and França, D. S. (2022). Faster quantum and classical SDP approximations for quadratic binary optimization. *Quantum*, **6**, 625. doi:10.22331/q-2022-01-20-625. arXiv:1909.04613.

[184] Brandhofer, S., Braun, D., Dehn, V., et al. (2023). Benchmarking the performance of portfolio optimization with QAOA. *Quantum Inf. Process.*, **22**, 1–27. doi:10.1007/s11128-022-03766-5. arXiv:2207.10555.

[185] van den Brand, J., Lee, Y. T., Sidford, A., and Song, Z. (2020). Solving tall dense linear programs in nearly linear time. Pages 775–788 of: *STOC*. doi:10.1145/3357713.3384309. arXiv:2002.02304.

[186] Brassard, G., Høyer, P., Mosca, M., and Tapp, A. (2002). Quantum amplitude amplification and estimation. Pages 53–74 of: *Quantum computation and quantum information: A millennium volume*, vol. 305. doi:10.1090/conm/305/05215. arXiv:quant-ph/0005055.

[187] Bravo-Prieto, C., LaRose, R., Cerezo, M., et al. (2019). Variational quantum linear solver. arXiv:1909.05820.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[188] Bravyi, S. (2015). Monte Carlo simulation of stoquastic Hamiltonians. *Quantum Inf. Comput.*, **15**, 1122–1140. doi:10.26421/QIC15.13-14-3. arXiv:1402.2295.

[189] Bravyi, S., and Cross, A. (2015). Doubled color codes. arXiv:1509.03239.

[190] Bravyi, S., and Kitaev, A. (2005). Universal quantum computation with ideal Clifford gates and noisy ancillas. *Phys. Rev. A*, **71**, 022316. doi:10.1103/physreva.71.022316. arXiv:quant-ph/0403025.

[191] Bravyi, S., and Terhal, B. (2010). Complexity of stoquastic frustration-free Hamiltonians. *SIAM J. Comp.*, **39**, 1462–1485. doi:10.1137/08072689X. arXiv:0806.1746.

[192] Bravyi, S., Suchara, M., and Vargo, A. (2014). Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A*, **90**, 032326. doi:10.1103/PhysRevA.90.032326. arXiv:1405.4883.

[193] Bravyi, S., Englbrecht, M., König, R., and Peard, N. (2018). Correcting coherent errors with surface codes. *npj Quant. Inf.*, **4**. doi:10.1038/s41534-018-0106-y.

[194] Bravyi, S., Kliesch, A., Koenig, R., and Tang, E. (2020). Obstacles to variational quantum optimization from symmetry protection. *Phys. Rev. Lett.*, **125**, 260505. doi:10.1103/PhysRevLett.125.260505. arXiv:1910.08980.

[195] Bravyi, S., Chowdhury, A., Gosset, D., and Wocjan, P. (2022). Quantum Hamiltonian complexity in thermal equilibrium. *Nat. Phys.*, **18**, 1367–1370. doi:10.1038/s41567-022-01742-5. arXiv:2110.15466.

[196] Bravyi, S. B., and Kitaev, A. Y. (1998). Quantum codes on a lattice with boundary. arXiv:quant-ph/9811052.

[197] Bravyi, S. B., and Kitaev, A. Y. (2002). Fermionic quantum computation. *Ann. Phys.*, **298**, 210–226. doi:10.1006/aphy.2002.6254. arXiv:quant-ph/0003137.

[198] Brenner, S. C., and Scott, L. R. (2008). *The mathematical theory of finite element methods*. Springer. doi:10.1007/978-0-387-75934-0.

[199] Breuckmann, N. P., and Eberhardt, J. N. (2021). Balanced product quantum codes. *IEEE Trans. Inf. Theory*, **67**, 6653–6674. doi:10.1109/tit.2021.3097347. arXiv:2012.09271.

[200] Breuckmann, N. P., Duivenvoorden, K., Michels, D., and Terhal, B. M. (2017). Local decoders for the 2D and 4D toric code. *Quantum Inf. Comput.*, **17**, 0181. doi:10.26421/QIC17.3-4-1. arXiv:1609.00510.

[201] Bridgeman, J. C., and Chubb, C. T. (2017). Hand-waving and interpretive dance: An introductory course on tensor networks. *J. Phys. A*, **50**, 223001. doi:10.1088/1751-8121/aa6dc3. arXiv:1603.03039.

[202] Bridgeman, J. C., Kubica, A., and Vasmer, M. (2024). Lifting topological codes: Three-dimensional subsystem codes from two-dimensional anyon models. *PRX Quantum*, **5**. doi:10.1103/prxquantum.5.020310. arXiv:2305.06365.

[203] Browaeys, A., and Lahaye, T. (2020). Many-body physics with individually controlled Rydberg atoms. *Nat. Phys.*, **16**, 132–142. doi:10.1038/s41567-019-0733-z. arXiv:2002.07413.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[204] Brown, A. R., Gharibyan, H., Leichenauer, S., et al. (2023). Quantum gravity in the lab. I. Teleportation by size and traversable wormholes. *PRX Quantum*, **4**, 010320. doi:10.1103/PRXQuantum.4.010320. arXiv:1911.06314.

[205] Brown, B. J. (2023). Conservation laws and quantum error correction: Towards a generalised matching decoder. *IEEE BITS Inf. Theory Mag.*, 1–12. doi:10.1109/MBITS.2023.3246025. arXiv:2207.06428.

[206] Brown, B. J., Laubscher, K., Kesselring, M. S., and Wootton, J. R. (2017). Poking holes and cutting corners to achieve Clifford gates with the surface code. *Phys. Rev. X*, **7**, 021029. doi:10.1103/PhysRevX.7.021029. arXiv:1609.04673.

[207] Brüstle, N., and Wiebe, N. (2024). Quantum and classical algorithms for nonlinear unitary dynamics. arXiv:2407.07685.

[208] Bruzewicz, C. D., Chiaverini, J., McConnell, R., and Sage, J. M. (2019). Trapped-ion quantum computing: Progress and challenges. *Appl. Phys. Rev.*, **6**, 021314. doi:10.1063/1.5088164. arXiv:1904.04178.

[209] Bădescu, C., and O'Donnell, R. (2021). Improved quantum data analysis. Pages 1398–1411 of: *STOC*. doi:10.1145/3406325.3451109. arXiv:2011.10908.

[210] Buhrman, H., Patro, S., and Speelman, F. (2021). A framework of quantum strong exponential-time hypotheses. Pages 19:1–19:19 of: *STACS*, vol. 187. doi:10.4230/LIPIcs.STACS.2021.19. arXiv:1911.05686.

[211] Bürgisser, P., Franks, C., Garg, A., et al. (2018). Efficient algorithms for tensor scaling, quantum marginals, and moment polytopes. Pages 883–897 of: *FOCS*. doi:10.1109/FOCS.2018.00088. arXiv:1804.04739.

[212] von Burg, V., Low, G. H., Häner, T., et al. (2021). Quantum computing enhanced computational catalysis. *Phys. Rev. Res.*, **3**, 033055. doi:10.1103/PhysRevResearch.3.033055. arXiv:2007.14460.

[213] Byrnes, T., and Yamamoto, Y. (2006). Simulating lattice gauge theories on a quantum computer. *Phys. Rev. A*, **73**, 022328. doi:10.1103/PhysRevA.73.022328. arXiv:quant-ph/0510027.

[214] Cade, C., and Crichigno, P. M. (2021). Complexity of supersymmetric systems and the cohomology problem. arXiv:2107.00011.

[215] Cade, C., Mineh, L., Montanaro, A., and Stanisic, S. (2020). Strategies for solving the Fermi–Hubbard model on near-term quantum computers. *Phys. Rev. B*, **102**, 235122. doi:10.1103/PhysRevB.102.235122. arXiv:1912.06007.

[216] Cade, C., Folkertsma, M., Niesen, I., and Weggemans, J. (2022). Quantum algorithms for community detection and their empirical run-times. arXiv:2203.06208.

[217] Cade, C., Folkertsma, M., Niesen, I., and Weggemans, J. (2023). Quantifying Grover speed-ups beyond asymptotic analysis. *Quantum*, **7**, 1133. doi:10.22331/q-2023-10-10-1133. arXiv:2203.04975.

[218] Cai, Z. (2020). Resource estimation for quantum variational simulations of the Hubbard model. *Phys. Rev. Appl.*, **14**, 014059. doi:10.1103/PhysRevApplied.14.014059. arXiv:1910.02719.

[219] Calabrese, P., and Cardy, J. (2005). Evolution of entanglement entropy in one-dimensional systems. *J. Stat. Mech. Theory Exp.*, 04010. doi:10.1088/1742-5468/2005/04/P04010. arXiv:cond-mat/0503393.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[220] Calabro, C., Impagliazzo, R., and Paturi, R. (2009). The complexity of satisfiability of small depth circuits. Pages 75–85 of: *Parameterized and Exact Computation*. doi:10.1007/978-3-642-11269-0_6.

[221] Calderbank, A. R., and Shor, P. W. (1996). Good quantum error-correcting codes exist. *Phys. Rev. A*, **54**, 1098–1105. doi:10.1103/physreva.54.1098. arXiv:quant-ph/9512032.

[222] Campbell, E. (2019). Random compiler for fast Hamiltonian simulation. *Phys. Rev. Lett.*, **123**. doi:10.1103/physrevlett.123.070503. arXiv:1811.08017.

[223] Campbell, E., Khurana, A., and Montanaro, A. (2019). Applying quantum algorithms to constraint satisfaction problems. *Quantum*, **3**, 167. doi:10.22331/q-2019-07-18-167. arXiv:1810.05582.

[224] Campbell, E. T. (2019). A theory of single-shot error correction for adversarial noise. *Quantum Sci. Technol.*, **4**, 025006. doi:10.1088/2058-9565/aafc8f. arXiv:1805.09271.

[225] Campbell, E. T. (2021). Early fault-tolerant simulations of the Hubbard model. *Quantum Sci. Technol.*, **7**, 015007. doi:10.1088/2058-9565/ac3110. arXiv:2012.09238.

[226] Campbell, E. T., Terhal, B. M., and Vuillot, C. (2017). Roads towards fault-tolerant universal quantum computation. *Nature*, **549**, 172–179. doi:10.1038/nature23460. arXiv:1612.07330.

[227] Camps, D., Lin, L., Van Beeumen, R., and Yang, C. (2024). Explicit quantum circuits for block encodings of certain sparse matrices. *SIAM J. Matrix Anal. Appl.*, **45**, 801–827. doi:10.1137/22M1484298. arXiv:2203.10236.

[228] Cao, Y., Papageorgiou, A., Petras, I., et al. (2013). Quantum algorithm and circuit design solving the poisson equation. *New J. Phys.*, **15**, 013021. doi:10.1088/1367-2630/15/1/013021. arXiv:1207.2485.

[229] Cao, Y., Romero, J., Olson, J. P., et al. (2019). Quantum chemistry in the age of quantum computing. *Chem. Rev.*, **119**, 10856–10915. doi:10.1021/acs.chemrev.8b00803. arXiv:1812.09976.

[230] Carbone, A., Cipollone, A., Barbieri, C., et al. (2013). Self-consistent Green's functions formalism with three-body interactions. *Phys. Rev. C*, **88**, 054326. doi:10.1103/PhysRevC.88.054326. arXiv:1310.3688.

[231] Carena, M., Lamm, H., Li, Y.-Y., and Liu, W. (2021). Lattice renormalization of quantum simulations. *Phys. Rev. D*, **104**, 094519. doi:10.1103/PhysRevD.104.094519. arXiv:2107.01166.

[232] Carlson, J., Gandolfi, S., Pederiva, F., et al. (2015). Quantum Monte Carlo methods for nuclear physics. *Rev. Mod. Phys.*, **87**, 1067. doi:10.1103/RevModPhys.87.1067. arXiv:1412.3081.

[233] Carlson, J., Savage, M. J., Gerber, R., et al. (2017). *Nuclear physics exascale requirements review: An Office of Science review sponsored jointly by Advanced Scientific Computing Research and Nuclear Physics, June 15–17, 2016, Gaithersburg, Maryland*. Tech. rept. OSTI 1906717. U.S. Department of Energy Office of Scientific and Technical Information. doi:10.2172/1369223.

[234] Caro, M. C., Gil-Fuster, E., Meyer, J. J., et al. (2021). Encoding-dependent generalization bounds for parametrized quantum circuits. *Quantum*, **5**, 582. doi:10.22331/q-2021-11-17-582. arXiv:2106.03880.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[235] Caro, M. C., Huang, H.-Y., Cerezo, M., et al. (2022). Generalization in quantum machine learning from few training data. *Nat. Commun.*, **13**, 4919. doi:10.1038/s41467-022-32550-3. arXiv:2111.05292.

[236] Caro, M. C., Huang, H.-Y., Ezzell, N., et al. (2023). Out-of-distribution generalization for learning quantum dynamics. *Nat. Commun.*, **14**, 3751. doi:10.1038/s41467-023-39381-w. arXiv:2204.10268.

[237] Carrington Jr, T. (2017). Perspective: Computing (ro-)vibrational spectra of molecules with more than four atoms. *J. Chem. Phys.*, **146**, 120902. doi:10.1063/1.4979117.

[238] Casares, P. A. M., Campos, R., and Martin-Delgado, M. A. (2022). TFermion: A non-Clifford gate cost assessment library of quantum phase estimation algorithms for quantum chemistry. *Quantum*, **6**, 768. doi:10.22331/q-2022-07-20-768. arXiv:2110.05899.

[239] Castryck, W., and Decru, T. (2023). An efficient key recovery attack on SIDH. Pages 423–447 of: *EUROCRYPT*. doi:10.1007/978-3-031-30589-4_15. ePrint:2022/975.

[240] Cerezo, M., Sone, A., Volkoff, T., et al. (2021). Cost function dependent barren plateaus in shallow parametrized quantum circuits. *Nat. Commun.*, **12**, 1–12. doi:10.1038/s41467-021-21728-w. arXiv:2001.00550.

[241] Cerezo, M., Arrasmith, A., Babbush, R., et al. (2021). Variational quantum algorithms. *Nat. Rev. Phys.*, 625–644. doi:10.1038/s42254-021-00348-9. arXiv:2012.09265.

[242] Cerezo, M., Verdon, G., Huang, H.-Y., et al. (2022). Challenges and opportunities in quantum machine learning. *Nat. Comput. Sci.*, **2**, 567–576. doi:10.1038/s43588-022-00311-3. arXiv:2303.09491.

[243] Cerezo, M., Larocca, M., García-Martín, D., et al. (2023). Does provable absence of barren plateaus imply classical simulability? Or, why we need to rethink variational quantum computing. arXiv:2312.09121.

[244] Cerjan, C. (1993). *Numerical grid methods and their application to Schrödinger's equation*. Vol. 412. Springer Science & Business Media. doi:10.1007/978-94-015-8240-7.

[245] Chakrabarti, S., Childs, A. M., Li, T., and Wu, X. (2020). Quantum algorithms and lower bounds for convex optimization. *Quantum*, **4**, 221. doi:10.22331/q-2020-01-13-221. arXiv:1809.01731.

[246] Chakrabarti, S., Krishnakumar, R., Mazzola, G., et al. (2021). A threshold for quantum advantage in derivative pricing. *Quantum*, **5**, 463. doi:10.22331/q-2021-06-01-463. arXiv:2012.03819.

[247] Chakrabarti, S., Minssen, P., Yalovetzky, R., and Pistoia, M. (2022). Universal quantum speedup for branch-and-bound, branch-and-cut, and tree-search algorithms. arXiv:2210.03210.

[248] Chakraborty, S., Gilyén, A., and Jeffery, S. (2019). The power of block-encoded matrix powers: Improved regression techniques via faster Hamiltonian simulation. Pages 33:1–33:14 of: *ICALP*. doi:10.4230/LIPIcs.ICALP.2019.33. arXiv:1804.01973.

[249] Chamberland, C., and Cross, A. W. (2019). Fault-tolerant magic state preparation with flag qubits. *Quantum*, **3**, 143. doi:10.22331/q-2019-05-20-143. arXiv:1811.00566.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[250] Chamberland, C., Noh, K., Arrangoiz-Arriola, P., et al. (2022). Building a fault-tolerant quantum computer using concatenated cat codes. *PRX Quantum*, **3**, 010329. doi:10.1103/PRXQuantum.3.010329. arXiv:2012.04108.

[251] Chamberland, C., Goncalves, L., Sivarajah, P., et al. (2023). Techniques for combining fast local decoders with global decoders under circuit-level noise. *Quantum Sci. Technol.*, **8**, 045011. doi:10.1088/2058-9565/ace64d. arXiv:2208.01178.

[252] Chan, G. K.-L. (2024). Spiers memorial lecture: Quantum chemistry, classical heuristics, and quantum advantage. *Faraday Discuss.*, **254**, 11–52. doi:10.1039/D4FD00141A. arXiv:2407.11235.

[253] Chan, H. H. S., Meister, R., Jones, T., et al. (2023). Grid-based methods for chemistry simulations on a quantum computer. *Sci. Adv.*, **9**, eabo7484. doi:10.1126/sciadv.abo7484. arXiv:2202.05864.

[254] Chao, R., and Reichardt, B. W. (2020). Flag fault-tolerant error correction for any stabilizer code. *PRX Quantum*, **1**. doi:10.1103/prxquantum.1.010302. arXiv:1912.09549.

[255] Chao, R., Ding, D., Gilyén, A., et al. (2020). Finding angles for quantum signal processing with machine precision. arXiv:2003.02831.

[256] Chen, C.-F., and Brandão, F. G. S. L. (2021). Fast thermalization from the eigenstate thermalization hypothesis. arXiv:2112.07646.

[257] Chen, C.-F., and Brandão, F. G. S. L. (2024). Average-case speedup for product formulas. *Commun. Math. Phys.*, **405**, 32. doi:10.1007/s00220-023-04912-5. arXiv:2111.05324.

[258] Chen, C.-F., Huang, H.-Y., Kueng, R., and Tropp, J. A. (2021). Concentration for random product formulas. *PRX Quantum*, **2**. doi:10.1103/prxquantum.2.040305. arXiv:2008.11751.

[259] Chen, C.-F., Kastoryano, M. J., and Gilyén, A. (2023). An efficient and exact noncommutative quantum Gibbs sampler. arXiv:2311.09207.

[260] Chen, C.-F., Kastoryano, M. J., Brandão, F. G. S. L., and Gilyén, A. (2023). Quantum thermal state preparation. arXiv:2303.18224.

[261] Chen, C.-F., Lucas, A., and Yin, C. (2023). Speed limits and locality in many-body quantum dynamics. *Rep. Prog. Phys.*, **86**, 116001. doi:10.1088/1361-6633/acfaae. arXiv:2303.07386.

[262] Chen, J., Stoudenmire, E., and White, S. R. (2023). Quantum Fourier transform has small entanglement. *PRX Quantum*, **4**, 040318. doi:10.1103/PRXQuantum.4.040318. arXiv:2210.08468.

[263] Chen, L., Jordan, S., Liu, Y.-K., et al. (2016). *Report on post-quantum cryptography*. Tech. rept. NISTIR 8105. National Institute of Standards and Technology, Gaithersburg, MD. doi:10.6028/NIST.IR.8105.

[264] Chen, M.-H., Yu, C.-H., Gao, J.-L., et al. (2022). Quantum algorithm for Gaussian process regression. *Phys. Rev. A*, **106**, 012406. doi:10.1103/PhysRevA.106.012406. arXiv:2106.06701.

[265] Chen, S., Cotler, J., Huang, H.-Y., and Li, J. (2022). Exponential separations between learning with and without quantum memory. Pages 574–585 of: *FOCS*. doi:10.1109/FOCS52979.2021.00063. arXiv:2111.05881.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[266] Chen, Y., and de Wolf, R. (2023). Quantum algorithms and lower bounds for linear regression with norm constraints. Pages 38:1–38:21 of: *ICALP*. doi:10.4230/LIPIcs.ICALP.2023.38. arXiv:2110.13086.

[267] Chen, Y., Gilyén, A., and de Wolf, R. (2025). A quantum speed-up for approximating the top eigenvectors of a matrix. In: *SODA*. Accepted, arXiv:2405.14765.

[268] Chen, Z.-Y., Xue, C., Chen, S.-M., et al. (2022). Quantum approach to accelerate finite volume method on steady computational fluid dynamics problems. *Quantum Inf. Process.*, **21**, 137. doi:10.1007/s11128-022-03478-w. arXiv:2102.03557.

[269] Chen, Z.-Y., Xue, C., Sun, T.-P., et al. (2023). An efficient and error-resilient protocol for quantum random access memory with generalized data size. arXiv:2303.05207.

[270] Chepurko, N., Clarkson, K., Horesh, L., et al. (2022). Quantum-inspired algorithms from randomized numerical linear algebra. Pages 3879–3900 of: *ICML*, vol. 162. arXiv:2011.04125.

[271] Chia, N.-H., Gilyén, A. P., Li, T., et al. (2022). Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning. *J. ACM*, **69**, 1–72. doi:10.1145/3549524. Earlier version in *STOC'20*. arXiv:1910.06151.

[272] Chiang, C.-F., and Wocjan, P. (2010). Quantum algorithm for preparing thermal Gibbs states-detailed analysis. Pages 138–147 of: *Quantum Cryptography and Computing*, vol. 26. doi:10.3233/978-1-60750-547-1-138. arXiv:1001.1130.

[273] Chiesa, A., Tacchino, F., Grossi, M., et al. (2019). Quantum hardware simulating four-dimensional inelastic neutron scattering. *Nat. Phys.*, **15**, 455–459. doi:10.1038/s41567-019-0437-4. arXiv:1809.07974.

[274] Childs, A., Jao, D., and Soukharev, V. (2014). Constructing elliptic curve isogenies in quantum subexponential time. *J. Math. Cryptol.*, **8**, 1–29. doi:10.1515/jmc-2012-0016. arXiv:1012.4019.

[275] Childs, A. M. (2010). On the relationship between continuous- and discrete-time quantum walk. *Commun. Math. Phys.*, **294**, 581–603. doi:10.1007/s00220-009-0930-1. arXiv:0810.0312.

[276] Childs, A. M. (2022). *Lecture notes on quantum algorithms*. https://cs.umd.edu/~amchilds/qa/, accessed: 2024-12-01.

[277] Childs, A. M., and van Dam, W. (2010). Quantum algorithms for algebraic problems. *Rev. Mod. Phys.*, **82**, 1–52. doi:10.1103/RevModPhys.82.1. arXiv:0812.0380.

[278] Childs, A. M., and Kothari, R. (2011). Simulating sparse Hamiltonians with star decompositions. Pages 94–103 of: *TQC*. doi:10.1007/978-3-642-18073-6_8. arXiv:1003.3683.

[279] Childs, A. M., and Liu, J.-P. (2020). Quantum spectral methods for differential equations. *Commun. Math. Phys.*, **375**, 1427–1457. doi:10.1007/s00220-020-03699-z. arXiv:1901.00961.

[280] Childs, A. M., and Su, Y. (2019). Nearly optimal lattice simulation by product formulas. *Phys. Rev. Lett.*, **123**, 050503. doi:10.1103/PhysRevLett.123.050503. arXiv:1901.00564.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[281] Childs, A. M., and Wiebe, N. (2012). Hamiltonian simulation using linear combinations of unitary operations. *Quantum Inf. Comput.*, **12**, 901–924. doi:10.26421/QIC12.11-12. arXiv:1202.5822.

[282] Childs, A. M., Kothari, R., and Somma, R. D. (2017). Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM J. Comp.*, **46**, 1920–1950. doi:10.1137/16M1087072. arXiv:1511.02306.

[283] Childs, A. M., Maslov, D., Nam, Y., et al. (2018). Toward the first quantum simulation with quantum speedup. *Proc. Natl. Acad. Sci.*, **115**, 9456–9461. doi:10.1073/pnas.1801723115. arXiv:1711.10980.

[284] Childs, A. M., Ostrander, A., and Su, Y. (2019). Faster quantum simulation by randomization. *Quantum*, **3**, 182. doi:10.22331/q-2019-09-02-182. arXiv:1805.08385.

[285] Childs, A. M., Liu, J.-P., and Ostrander, A. (2021). High-precision quantum algorithms for partial differential equations. *Quantum*, **5**, 574. doi:10.22331/q-2021-11-10-574. arXiv:2002.07868.

[286] Childs, A. M., Su, Y., Tran, M. C., et al. (2021). Theory of Trotter error with commutator scaling. *Phys. Rev. X*, **11**. doi:10.1103/physrevx.11.011020. arXiv:1912.08854.

[287] Childs, A. M., Leng, J., Li, T., et al. (2022). Quantum simulation of real-space dynamics. *Quantum*, **6**, 860. doi:10.22331/q-2022-11-17-860. arXiv:2203.17006.

[288] Cho, C. H., Berry, D. W., and Hsieh, M.-H. (2022). Doubling the order of approximation via the randomized product formula. arXiv:2210.11281.

[289] Chowdhury, A. N., and Somma, R. D. (2017). Quantum algorithms for Gibbs sampling and hitting-time estimation. *Quantum Inf. Comput.*, **17**, 41–64. doi:10.26421/QIC17.1-2. arXiv:1603.02940.

[290] Chowdhury, A. N., Somma, R. D., and Subaşı, Y. (2021). Computing partition functions in the one-clean-qubit model. *Phys. Rev. A*, **103**, 032422. doi:10.1103/PhysRevA.103.032422. arXiv:1910.11842.

[291] Chubb, C. T. (2021). General tensor network decoding of 2D Pauli codes. arXiv:2101.04125.

[292] Ciavarella, A., Klco, N., and Savage, M. J. (2021). Trailhead for quantum simulation of SU(3) Yang–Mills lattice gauge theory in the local multiplet basis. *Phys. Rev. D*, **103**, 094501. doi:10.1103/PhysRevD.103.094501. arXiv:2101.10227.

[293] Ciliberto, C., Herbster, M., Ialongo, A. D., et al. (2018). Quantum machine learning: A classical perspective. *Proc. R. Soc. A*, **474**, 20170551. doi:10.1098/rspa.2017.0551. arXiv:1707.08561.

[294] Cirstoiu, C., Holmes, Z., Iosue, J., et al. (2020). Variational fast forwarding for quantum simulation beyond the coherence time. *npj Quant. Inf.*, **6**, 82. doi:10.1038/s41534-020-00302-0. arXiv:1910.04292.

[295] Clader, B. D., Jacobs, B. C., and Sprouse, C. R. (2013). Preconditioned quantum linear system algorithm. *Phys. Rev. Lett.*, **110**, 250504. doi:10.1103/PhysRevLett.110.250504. arXiv:1301.2340.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[296] Clader, B. D., Dalzell, A. M., Stamatopoulos, N., et al. (2022). Quantum resources required to block-encode a matrix of classical data. *IEEE Trans. Quantum Eng.*, **3**, 1–23. doi:10.1109/TQE.2022.3231194. arXiv:2206.03505.

[297] Cleve, R., and Wang, C. (2017). Efficient quantum algorithms for simulating Lindblad evolution. Pages 17:1–17:14 of: *ICALP*. doi:10.4230/LIPIcs.ICALP.2017.17. arXiv:1612.09512.

[298] Cleve, R., and Watrous, J. (2000). Fast parallel circuits for the quantum Fourier transform. Pages 526–536 of: *FOCS*. doi:10.1109/SFCS.2000.892140. arXiv:quant-ph/0006004.

[299] Cleve, R., Ekert, A., Macchiavello, C., and Mosca, M. (1998). Quantum algorithms revisited. *Proc. R. Soc. A*, **454**, 339–354. doi:10.1098/rspa.1998.0164. arXiv:quant-ph/9708016.

[300] Clinton, L., Bausch, J., and Cubitt, T. (2021). Hamiltonian simulation algorithms for near-term quantum hardware. *Nat. Commun.*, **12**, 4989. doi:10.1038/s41467-021-25196-0. arXiv:2003.06886.

[301] Clinton, L., Cubitt, T., Flynn, B., et al. (2024). Towards near-term quantum simulation of materials. *Nat. Commun.*, **15**, 211. doi:10.1038/s41467-023-43479-6. arXiv:2205.15256.

[302] Coecke, B., and Kissinger, A. (2017). *Picturing quantum processes*. Cambridge University Press. doi:10.1017/9781316219317.

[303] Cohen, L. Z., Kim, I. H., Bartlett, S. D., and Brown, B. J. (2022). Low-overhead fault-tolerant quantum computing using long-range connectivity. *Sci. Adv.*, **8**. doi:10.1126/sciadv.abn1717.

[304] Cohen, M. B., Lee, Y. T., and Song, Z. (2021). Solving linear programs in the current matrix multiplication time. *J. ACM*, **68**. doi:10.1145/3424305. arXiv:1810.07896.

[305] Commeau, B., Cerezo, M., Holmes, Z., et al. (2020). Variational Hamiltonian diagonalization for dynamical quantum simulation. arXiv:2009.02559.

[306] Cong, I., Choi, S., and Lukin, M. D. (2019). Quantum convolutional neural networks. *Nat. Phys.*, **15**, 1273–1278. doi:10.1038/s41567-019-0648-8. arXiv:1810.03787.

[307] Coppersmith, D., Gamarnik, D., Hajiaghayi, M., and Sorkin, G. B. (2004). Random MAX SAT, random MAX CUT, and their phase transitions. *Rand. Struct. Algorithms*, **24**, 502–545. doi:10.1002/rsa.20015. Earlier version in *SODA'03*. arXiv:math/0306047.

[308] Cornelissen, A., and Hamoudi, Y. (2023). A sublinear-time quantum algorithm for approximating partition functions. Pages 1245–1264 of: *SODA*. doi:10.1137/1.9781611977554.ch46. arXiv:2207.08643.

[309] Cornelissen, A., and Jerbi, S. (2021). Quantum algorithms for multivariate Monte Carlo estimation. arXiv:2107.03410.

[310] Cornelissen, A., Hamoudi, Y., and Jerbi, S. (2022). Near-optimal quantum algorithms for multivariate mean estimation. Pages 33–43 of: *STOC*. doi:10.1145/3519935.3520045. arXiv:2111.09787.

[311] Cornuejols, G., and Tütüncü, R. (2006). *Optimization methods in finance*. Vol. 5. Cambridge University Press.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[312] Costa, P. C., Jordan, S., and Ostrander, A. (2019). Quantum algorithm for simulating the wave equation. *Phys. Rev. A*, **99**, 012323. doi:10.1103/PhysRevA.99.012323.

[313] Costa, P. C., An, D., Sanders, Y. R., et al. (2022). Optimal scaling quantum linear-systems solver via discrete adiabatic theorem. *PRX Quantum*, **3**, 040303. doi:10.1103/PRXQuantum.3.040303. arXiv:2111.08152.

[314] Costa, P. C. S., An, D., Babbush, R., and Berry, D. (2023). The discrete adiabatic quantum linear system solver has lower constant factors than the randomized adiabatic solver. arXiv:2312.07690.

[315] Costa, P. C. S., Schleich, P., Morales, M. E. S., and Berry, D. W. (2023). Further improving quantum algorithms for nonlinear differential equations via higher-order methods and rescaling. arXiv:2312.09518.

[316] Cotler, J. S., Gur-Ari, G., Hanada, M., et al. (2017). Black holes and random matrices. *J. High Energy Phys.*, **2017**, 1–54. doi:10.1007/JHEP05(2017)118. arXiv:1611.04650.

[317] Cramer, M., Plenio, M. B., Flammia, S. T., et al. (2010). Efficient quantum state tomography. *Nat. Commun.*, **1**, 149. doi:10.1038/ncomms1147. arXiv:1101.4366.

[318] Cramer, R., Ducas, L., and Wesolowski, B. (2017). Short Stickelberger class relations and application to Ideal-SVP. Pages 324–348 of: *EUROCRYPT*. ePrint:2016/885.

[319] Crichigno, M., and Kohler, T. (2022). Clique homology is QMA1-hard. arXiv:2209.11793.

[320] Criger, B., and Ashraf, I. (2018). Multi-path summation for decoding 2d topological codes. *Quantum*, **2**, 102. doi:10.22331/q-2018-10-19-102. arXiv:1709.02154.

[321] Crooks, G. E. (2019). Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition. arXiv:1905.13311.

[322] Crosson, E., and Harrow, A. W. (2016). Simulated quantum annealing can be exponentially faster than classical simulated annealing. Pages 714–723 of: *FOCS*. doi:10.1109/FOCS.2016.81. arXiv:1601.03030.

[323] Crosson, E., and Lidar, D. (2021). Prospects for quantum enhancement with diabatic quantum annealing. *Nat. Rev. Phys.*, **3**, 466–489. doi:10.1038/s42254-021-00313-6. arXiv:2008.09913.

[324] Crosson, E., and Slezak, S. (2020). Classical simulation of high temperature quantum Ising models. doi:10.48550/arXiv.2002.02232. arXiv:2002.02232.

[325] Cunningham, J., and Roland, J. (2024). Eigenpath traversal by poisson-distributed phase randomisation. arXiv:2406.03972.

[326] Daley, A. J., Bloch, I., Kokail, C., et al. (2022). Practical quantum advantage in quantum simulation. *Nature*, **607**, 667–676. doi:10.1038/s41586-022-04940-6.

[327] Dalzell, A. M. (2024). A shortcut to an optimal quantum linear system solver. arXiv:2406.12086.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[328] Dalzell, A. M., Clader, B. D., Salton, G., et al. (2023). End-to-end resource analysis for quantum interior-point methods and portfolio optimization. *PRX Quantum*, **4**, 040325. doi:10.1103/PRXQuantum.4.040325. arXiv:2211.12489.

[329] Dalzell, A. M., Pancotti, N., Campbell, E. T., and Brandão, F. G. (2023). Mind the gap: Achieving a super-Grover quantum speedup by jumping to the end. Pages 1131–1144 of: *STOC*. doi:10.1145/3564246.3585203. arXiv:2212.01513.

[330] Damm, C., Holzer, M., and McKenzie, P. (2002). The complexity of tensor calculus. *Comput. Complex.*, **11**, 54–89. doi:10.1007/s00037-000-0170-4.

[331] Darmawan, A. S., and Poulin, D. (2017). Tensor-network simulations of the surface code under realistic noise. *Phys. Rev. Lett.*, **119**, 040502. doi:10.1103/PhysRevLett.119.040502. arXiv:1607.06460.

[332] Davoudi, Z., Detmold, W., Shanahan, P., et al. (2021). Nuclear matrix elements from lattice QCD for electroweak and beyond-Standard-Model processes. *Phys. Rep.*, **900**, 1–74. doi:10.1016/j.physrep.2020.10.004.

[333] Davoudi, Z., Neil, E. T., Bauer, C. W., et al. (2022). Report of the Snowmass 2021 topical group on lattice gauge theory. *Proceedings of the US Community Study on the Future of Particle Physics (Snowmass 2021)*. arXiv:2209.10758.

[334] Davoudi, Z., Shaw, A. F., and Stryker, J. R. (2023). General quantum algorithms for Hamiltonian simulation with applications to a non-abelian lattice gauge theory. *Quantum*, **7**, 1213. doi:10.22331/q-2023-12-20-1213. arXiv:2212.14030.

[335] de Beaudrap, N., and Horsman, D. (2020). The ZX calculus is a language for surface code lattice surgery. *Quantum*, **4**, 218. doi:10.22331/q-2020-01-09-218. arXiv:1704.08670.

[336] De Palma, G., Marvian, M., Rouzé, C., and França, D. S. (2023). Limitations of variational quantum algorithms: A quantum optimal transport approach. *PRX Quantum*, **4**, 010309. doi:10.1103/PRXQuantum.4.010309. arXiv:2204.03455.

[337] Dean, D. J. (2007). Beyond the nuclear shell model. *Phys. Today*, **60**, 48–53. doi:10.1063/1.2812123.

[338] Delfosse, N., and Nickerson, N. H. (2021). Almost-linear time decoding algorithm for topological codes. *Quantum*, **5**, 595. doi:10.22331/q-2021-12-02-595. arXiv:1709.06218.

[339] Delfosse, N., Reichardt, B. W., and Svore, K. M. (2022). Beyond single-shot fault-tolerant quantum error correction. *IEEE Trans. Inf. Theory*, **68**, 287–301. doi:10.1109/tit.2021.3120685. arXiv:2002.05180.

[340] Delgado, A., Hamilton, K. E., Date, P., et al. (2022). Quantum computing for data analysis in high-energy physics. arXiv:2203.08805.

[341] Delgado, A., Casares, P. A. M., dos Reis, R., et al. (2022). Simulating key properties of lithium-ion batteries with a fault-tolerant quantum computer. *Phys. Rev. A*, **106**, 032428. doi:10.1103/PhysRevA.106.032428. arXiv:2204.11890.

[342] Deng, C., Sun, F., Qian, X., et al. (2019). TIE: Energy-efficient tensor train-based inference engine for deep neural network. Pages 264–278 of: *ISCA*. doi:10.1145/3307650.3322258.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[343] Dennis, E., Kitaev, A., Landahl, A., and Preskill, J. (2002). Topological quantum memory. *J. Math. Phys.*, **43**, 4452–4505. doi:10.1063/1.1499754. arXiv:quant-ph/0110143.

[344] Derby, C., Klassen, J., Bausch, J., and Cubitt, T. (2021). Compact fermion to qubit mappings. *Phys. Rev. B*, **104**, 035118. doi:10.1103/PhysRevB.104.035118. arXiv:2003.06939.

[345] Derrida, B. (1980). Random-energy model: Limit of a family of disordered models. *Phys. Rev. Lett.*, **45**, 79–82. doi:10.1103/PhysRevLett.45.79.

[346] Deutsch, D. (1985). Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. A*, **400**, 97–117. doi:10.1098/rspa.1985.0070.

[347] Deutsch, D., and Jozsa, R. (1992). Rapid solution of problems by quantum computation. *Proc. R. Soc. A*, **439**, 553–558. doi:10.1098/rspa.1992.0167.

[348] Devoret, M. H., and Schoelkopf, R. J. (2013). Superconducting circuits for quantum information: An outlook. *Science*, **339**, 1169–1174. doi:10.1126/science.123193.

[349] Di Matteo, O., Gheorghiu, V., and Mosca, M. (2020). Fault-tolerant resource estimation of quantum random-access memories. *IEEE Trans. Quantum Eng.*, **1**, 1–13. doi:10.1109/TQE.2020.2965803. arXiv:1902.01329.

[350] Ding, C., Bao, T.-Y., and Huang, H.-L. (2022). Quantum-inspired support vector machine. *IEEE Trans. Neural Netw. Learn. Syst.*, **33**, 7210–7222. doi:10.1109/TNNLS.2021.3084467. arXiv:1906.08902.

[351] Ding, Z., Chen, C.-F., and Lin, L. (2024). Single-ancilla ground state preparation via Lindbladians. *Phys. Rev. Res.*, **6**, 033147. doi:10.1103/PhysRevResearch.6.033147. arXiv:2308.15676.

[352] Dinur, I., Evra, S., Livne, R., et al. (2022). Locally testable codes with constant rate, distance, and locality. Pages 357–374 of: *STOC*. doi:10.1145/3519935.3520024. arXiv:2111.04808.

[353] DiVincenzo, D. P., and Aliferis, P. (2007). Effective fault-tolerant quantum computation with slow measurements. *Phys. Rev. Lett.*, **98**. doi:10.1103/physrevlett.98.020501.

[354] Dodin, I. Y., and Startsev, E. A. (2021). On applications of quantum computing to plasma simulations. *Phys. Plasmas*, **28**, 092101. doi:10.1063/5.0056974. arXiv:2005.14369.

[355] Domahidi, A., Chu, E., and Boyd, S. (2013). ECOS: An SOCP solver for embedded systems. Pages 3071–3076 of: *ECC*. doi:10.23919/ECC.2013.6669541.

[356] Dong, Y., Meng, X., Whaley, K. B., and Lin, L. (2021). Efficient phase-factor evaluation in quantum signal processing. *Phys. Rev. A*, **103**, 042419. doi:10.1103/PhysRevA.103.042419. arXiv:2002.11649.

[357] Dong, Y., Gross, J., and Niu, M. Y. (2022). Beyond Heisenberg limit quantum metrology through quantum signal processing. arXiv:2209.11207.

[358] Dong, Y., Lin, L., and Tong, Y. (2022). Ground-state preparation and energy estimation on early fault-tolerant quantum computers via quantum eigenvalue transformation of unitary matrices. *PRX Quantum*, **3**, 040305. doi:10.1103/PRXQuantum.3.040305. arXiv:2204.05955.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[359] Dong, Y., Lin, L., Ni, H., and Wang, J. (2022). Infinite quantum signal processing. arXiv:2209.10162.

[360] Dong, Y., Lin, L., Ni, H., and Wang, J. (2023). Robust iterative method for symmetric quantum signal processing in all parameter regimes. arXiv:2307.12468.

[361] Doriguello, J. a. F., Luongo, A., Bao, J., et al. (2022). Quantum algorithm for stochastic optimal stopping problems with applications in finance. Pages 2:1–2:24 of: *TQC*. doi:10.4230/LIPIcs.TQC.2022.2. arXiv:2111.15332.

[362] Du, Y., Hsieh, M.-H., Liu, T., and Tao, D. (2020). Expressive power of parametrized quantum circuits. *Phys. Rev. Res.*, **2**, 033125. doi:10.1103/PhysRevResearch.2.033125. arXiv:1810.11922.

[363] Dua, A., Kubica, A., Jiang, L., et al. (2024). Clifford-deformed surface codes. *PRX Quantum*, **5**. doi:10.1103/prxquantum.5.010347. arXiv:2201.07802.

[364] Duclos-Cianci, G., and Poulin, D. (2010). Fast decoders for topological quantum codes. *Phys. Rev. Lett.*, **104**, 050504. doi:10.1103/PhysRevLett.104.050504. arXiv:0911.0581.

[365] Dukalski, M. (2021). Toward an application of quantum computing in geophysics. Pages 1–5 of: *Fifth EAGE Workshop on High Performance Computing for Upstream*, vol. 2021. doi:10.3997/2214-4609.2021612005.

[366] Dumitrescu, E. F., McCaskey, A. J., Hagen, G., et al. (2018). Cloud quantum computing of an atomic nucleus. *Phys. Rev. Lett.*, **120**, 210501. doi:10.1103/PhysRevLett.120.210501. arXiv:1801.03897.

[367] Dürr, C., and Høyer, P. (1996). A quantum algorithm for finding the minimum. arXiv:quant-ph/9607014.

[368] Dürr, C., Heiligman, M., Høyer, P., and Mhalla, M. (2006). Quantum query complexity of some graph problems. *SIAM J. Comp.*, **35**, 1310–1328. doi:10.1137/050644719. Earlier version in *ICALP'04*. arXiv:quant-ph/0401091.

[369] Dürr, S., Fodor, Z., Frison, J., et al. (2008). Ab initio determination of light hadron masses. *Science*, **322**, 1224–1227. doi:10.1126/science.1163233. arXiv:0906.3599.

[370] Eastin, B., and Knill, E. (2009). Restrictions on transversal encoded quantum gate sets. *Phys. Rev. Lett.*, **102**, 110502. doi:10.1103/physrevlett.102.110502. arXiv:0811.4262.

[371] Ebadi, S., Wang, T. T., Levine, H., et al. (2021). Quantum phases of matter on a 256-atom programmable quantum simulator. *Nature*, **595**, 227. doi:10.1038/s41586-021-03582-4. arXiv:2012.12281.

[372] Edmonds, J. (1965). Paths, trees, and flowers. *Can. J. Math.*, **17**, 449–467. doi:10.4153/CJM-1965-045-4.

[373] Elben, A., Flammia, S. T., Huang, H.-Y., et al. (2023). The randomized measurement toolbox. *Nat. Rev. Phys.*, **5**, 9–24. doi:10.1038/s42254-022-00535-2. arXiv:2203.11374.

[374] Elenewski, J. E., Camara, C. M., and Kalev, A. (2024). Prospects for NMR spectral prediction on fault-tolerant quantum computers. arXiv:2406.09340.

[375] Elfving, V. E., Broer, B. W., Webber, M., et al. (2020). How will quantum computers provide an industrially relevant computational advantage in quantum chemistry? arXiv:2009.12472.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[376] Elgart, A., and Hagedorn, G. A. (2012). A note on the switching adiabatic theorem. *J. Math. Phys.*, **53**, 102202. doi:10.1063/1.4748968. arXiv:1204.2318.

[377] Engel, A., Smith, G., and Parker, S. E. (2019). Quantum algorithm for the Vlasov equation. *Phys. Rev. A*, **100**, 062315. doi:10.1103/PhysRevA.100.062315. arXiv:1907.09418.

[378] Engineering, S., and (SEAR), A. (2024). *iMessage with PQ3: The new state of the art in quantum-secure messaging at scale*. https://security.apple.com/blog/imessage-pq3/, accessed: 2024-06-05.

[379] Fang, D., Lin, L., and Tong, Y. (2023). Time-marching based quantum solvers for time-dependent linear differential equations. *Quantum*, **7**, 955. doi:10.22331/q-2023-03-20-955. arXiv:2208.06941.

[380] Farhi, E., and Harrow, A. W. (2016). Quantum supremacy through the quantum approximate optimization algorithm. arXiv:1602.07674.

[381] Farhi, E., and Neven, H. (2018). Classification with quantum neural networks on near term processors. arXiv:1802.06002.

[382] Farhi, E., Goldstone, J., Gutmann, S., and Sipser, M. (2000). Quantum computation by adiabatic evolution. arXiv:quant-ph/0001106.

[383] Farhi, E., Goldstone, J., Gosset, D., et al. (2009). Quantum adiabatic algorithms, small gaps, and different paths. *Quantum Inf. Comput.*, **11**, 181–214. doi:10.26421/QIC11.3-4-1. arXiv:0909.4766.

[384] Farhi, E., Goldstone, J., and Gutmann, S. (2014). A quantum approximate optimization algorithm. arXiv:1411.4028.

[385] Farhi, E., Gamarnik, D., and Gutmann, S. (2020). The quantum approximate optimization algorithm needs to see the whole graph: A typical case. arXiv:2004.09002.

[386] Farhi, E., Gamarnik, D., and Gutmann, S. (2020). The quantum approximate optimization algorithm needs to see the whole graph: Worst case examples. arXiv:2005.08747.

[387] Farrell, R. C., Illa, M., Ciavarella, A. N., and Savage, M. J. (2024). Quantum simulations of hadron dynamics in the Schwinger model using 112 qubits. arXiv:2401.08044.

[388] Fawzi, O., Grospellier, A., and Leverrier, A. (2018). Constant overhead quantum fault-tolerance with quantum expander codes. In: *FOCS*. doi:10.1109/focs.2018.00076. arXiv:1808.03821.

[389] Felser, T., Silvi, P., Collura, M., and Montangero, S. (2020). Two-dimensional quantum-link lattice quantum electrodynamics at finite density. *Phys. Rev. X*, **10**, 041040. doi:10.1103/PhysRevX.10.041040. arXiv:1911.09693.

[390] Felser, T., Trenti, M., Sestini, L., et al. (2021). Quantum-inspired machine learning on high-energy physics data. *npj Quant. Inf.*, **7**, 111. doi:10.1038/s41534-021-00443-w. arXiv:2004.13747.

[391] Feynman, R. P. (1982). Simulating physics with computers. *Int. J. Th. Phys.*, **21**, 467–488. doi:10.1007/BF02650179.

[392] Flannigan, S., Pearson, N., Low, G. H., et al. (2022). Propagation of errors and quantitative quantum simulation with quantum advantage. *Quantum Sci. Technol.*, **7**, 045025. doi:10.1088/2058-9565/ac88f5. arXiv:2204.13644.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[393] Focardi, S., Fabozzi, F. J., and Mazza, D. (2020). Quantum option pricing and quantum finance. *J. Deriv.*, **28**, 79–98. doi:10.3905/jod.2020.1.111.

[394] Fomichev, S., Hejazi, K., Zini, M. S., et al. (2024). Initial state preparation for quantum chemistry on quantum computers. *PRX Quantum*, **5**, 040339. doi:10.1103/PRXQuantum.5.040339. arXiv:2310.18410.

[395] Fomichev, S., Hejazi, K., Loaiza, I., et al. (2024). Simulating X-ray absorption spectroscopy of battery materials on a quantum computer. arXiv:2405.11015.

[396] Fontana, E., Herman, D., Chakrabarti, S., et al. (2024). Characterizing barren plateaus in quantum ansätze with the adjoint representation. *Nat. Commun.*, **15**, 7171. doi:10.1038/s41467-024-49910-w. arXiv:2309.07902.

[397] Fowler, A. G., and Gidney, C. (2018). Low overhead quantum computation using lattice surgery. arXiv:1808.06709.

[398] Fowler, A. G., Mariantoni, M., Martinis, J. M., and Cleland, A. N. (2012). Surface codes: Towards practical large-scale quantum computation. *Phys. Rev. A*, **86**, 032324. doi:10.1103/PhysRevA.86.032324. arXiv:1208.0928.

[399] Fradkin, E., Kivelson, S. A., and Tranquada, J. M. (2015). Colloquium: Theory of intertwined orders in high temperature superconductors. *Rev. Mod. Phys.*, **87**, 457–482. doi:10.1103/RevModPhys.87.457. arXiv:1407.4480.

[400] França, D. S., and Garcia-Patron, R. (2021). Limitations of optimization algorithms on noisy quantum devices. *Nat. Phys.*, **17**, 1221–1227. doi:10.1038/s41567-021-01356-3. arXiv:2009.05532.

[401] Franz, M., and Rozali, M. (2018). Mimicking black hole event horizons in atomic and solid-state systems. *Nat. Rev. Mater.*, **3**, 491–501. doi:10.1038/s41578-018-0058-z. arXiv:1808.00541.

[402] Fraxanet, J., Salamon, T., and Lewenstein, M. (2023). The coming decades of quantum simulation. Pages 85–125 of: *Sketches of Physics: The Celebration Collection*. doi:10.1007/978-3-031-32469-7_4. arXiv:2204.08905.

[403] Friedman, J. (1998). Computing Betti numbers via combinatorial Laplacians. *Algorithmica*, **21**, 331–346. doi:10.1007/PL00009218.

[404] Funcke, L., Hartung, T., Jansen, K., and Kühn, S. (2023). Review on quantum computing for lattice field theory. Page 228 of: *LATTICE*, vol. 430. doi:10.22323/1.430.0228. arXiv:2302.00467.

[405] Gaitan, F. (2020). Finding flows of a Navier–Stokes fluid through quantum computing. *npj Quant. Inf.*, **6**, 61. doi:10.1038/s41534-020-00291-0.

[406] Gaitan, F. (2021). Finding solutions of the Navier–Stokes equations through quantum computing—recent progress, a generalization, and next steps forward. *Adv. Quantum Technol.*, **4**, 2100055. doi:10.1002/qute.202100055.

[407] Garcia-Álvarez, L., Egusquiza, I. L., Lamata, L., et al. (2017). Digital quantum simulation of minimal AdS/CFT. *Phys. Rev. Lett.*, **119**, 040501. doi:10.1103/PhysRevLett.119.040501. arXiv:1607.08560.

[408] García-García, A. M., and Verbaarschot, J. J. M. (2016). Spectral and thermodynamic properties of the Sachdev–Ye–Kitaev model. *Phys. Rev. D*, **94**, 126010. doi:10.1103/PhysRevD.94.126010. arXiv:1610.03816.

[409] Garg, A., Kothari, R., Netrapalli, P., and Sherif, S. (2021). Near-optimal lower bounds for convex optimization for all orders of smoothness. Pages 29874–29884 of: *NeurIPS*, vol. 34. arXiv:2112.01118.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[410] Garg, A., Kothari, R., Netrapalli, P., and Sherif, S. (2021). No quantum speedup over gradient descent for non-smooth convex optimization. Pages 53:1–53:20 of: *ITCS*, vol. 185. doi:10.4230/LIPIcs.ITCS.2021.53. arXiv:2010.01801.

[411] Ge, Y., Molnár, A., and Cirac, J. I. (2016). Rapid adiabatic preparation of injective projected entangled pair states and Gibbs states. *Phys. Rev. Lett.*, **116**, 080503. doi:10.1103/PhysRevLett.116.080503. arXiv:1508.00570.

[412] Ge, Y., Tura, J., and Cirac, J. I. (2019). Faster ground state preparation and high-precision ground energy estimation with fewer qubits. *J. Math. Phys.*, **60**, 022202. doi:10.1063/1.5027484. arXiv:1712.03193.

[413] Gehrmann, T., and Malaescu, B. (2022). Precision QCD physics at the LHC. *Annu. Rev. Nucl. Part. Sci.*, **72**, 233–258. doi:10.1146/annurev-nucl-101920-014923. arXiv:2111.02319.

[414] Gentinetta, G., Sutter, D., Zoufal, C., et al. (2023). Quantum kernel alignment with stochastic gradient descent. Pages 256–262 of: *QCE*, vol. 01. doi:10.1109/QCE57702.2023.00036. arXiv:2304.09899.

[415] Gentinetta, G., Thomsen, A., Sutter, D., and Woerner, S. (2024). The complexity of quantum support vector machines. *Quantum*, **8**, 1225. doi:10.22331/q-2024-01-11-1225. arXiv:2203.00031.

[416] Georges, T. N., Bothe, M., Sünderhauf, C., et al. (2024). Quantum simulations of chemistry in first quantization with any basis set. arXiv:2408.03145.

[417] Georgescu, I. M., Ashhab, S., and Nori, F. (2014). Quantum simulation. *Rev. Mod. Phys.*, **86**, 153–185. doi:10.1103/RevModPhys.86.153. arXiv:1308.6253.

[418] Gharibian, S. (2024). *Quantum complexity theory*. https://groups.uni-paderborn.de/fg-qi/data/QCT_Masterfile.pdf, accessed: 2024-11-19.

[419] Gharibian, S., and Le Gall, F. (2022). Dequantizing the quantum singular value transformation: Hardness and applications to quantum chemistry and the quantum PCP conjecture. Pages 19–32 of: *STOC*. doi:10.1145/3519935.3519991. arXiv:2111.09079.

[420] Gibbs, J., Gili, K., Holmes, Z., et al. (2022). Long-time simulations for fixed input states on quantum hardware. *npj Quant. Inf.*, **8**, 135. doi:10.1038/s41534-022-00625-0. arXiv:2102.04313.

[421] Gidney, C. (2018). Halving the cost of quantum addition. *Quantum*, **2**, 74. doi:10.22331/q-2018-06-18-74. arXiv:1709.06648.

[422] Gidney, C. (2021). Stim: A fast stabilizer circuit simulator. *Quantum*, **5**, 497. doi:10.22331/q-2021-07-06-497. arXiv:2103.02202.

[423] Gidney, C. (2024). Inplace access to the surface code Y basis. *Quantum*, **8**, 1310. doi:10.22331/q-2024-04-08-1310. arXiv:2302.07395.

[424] Gidney, C., and Ekerå, M. (2021). How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. *Quantum*, **5**, 433. doi:10.22331/q-2021-04-15-433. arXiv:1905.09749.

[425] Gil-Fuster, E., Eisert, J., and Dunjko, V. (2023). On the expressivity of embedding quantum kernels. *Mach. Learn.: Sci. Technol.*, **5**, 025003. doi:10.1088/2632-2153/ad2f51. arXiv:2309.14419.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[426] Giles, M. B. (2015). Multilevel Monte Carlo methods. *Acta Numer.*, **24**, 259–328. doi:10.1017/S096249291500001X. arXiv:1304.5472.

[427] Gilyén, A. (2014). *Quantum walk based search methods and algorithmic applications*. Master's thesis, Eötvös Loránd University.

[428] Gilyén, A. (2019). *Quantum singular value transformation & its algorithmic applications*. PhD thesis, University of Amsterdam.

[429] Gilyén, A., Su, Y., Low, G. H., and Wiebe, N. (2018). Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics [full version]. arXiv:1806.01838.

[430] Gilyén, A., Arunachalam, S., and Wiebe, N. (2019). Optimizing quantum optimization algorithms via faster quantum gradient computation. Pages 1425–1444 of: *SODA*. doi:10.1137/1.9781611975482.87. arXiv:1711.00465.

[431] Gilyén, A., Su, Y., Low, G. H., and Wiebe, N. (2019). Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics. Pages 193–204 of: *STOC*. doi:10.1145/3313276.3316366. arXiv:1806.01838.

[432] Gilyén, A., Hastings, M. B., and Vazirani, U. (2021). (Sub)Exponential advantage of adiabatic quantum computation with no sign problem. Pages 1357–1369 of: *STOC*. doi:10.1145/3406325.3451060. arXiv:2011.09495.

[433] Gilyén, A., Song, Z., and Tang, E. (2022). An improved quantum-inspired algorithm for linear regression. *Quantum*, **6**, 754. doi:10.22331/q-2022-06-30-754. arXiv:2009.07268.

[434] Giovannetti, V., Lloyd, S., and Maccone, L. (2008). Quantum random access memory. *Phys. Rev. Lett.*, **100**, 160501. doi:10.1103/PhysRevLett.100.160501. arXiv:0708.1879.

[435] Glasserman, P. (2004). *Monte Carlo methods in financial engineering*. Vol. 53. Springer. doi:10.1007/978-0-387-21617-1.

[436] Glick, J. R., Gujarati, T. P., Córcoles, A. D., et al. (2024). Covariant quantum kernels for data with group structure. *Nat. Phys.*, **20**, 479–483. doi:10.1038/s41567-023-02340-9. arXiv:2105.03406.

[437] Goings, J. J., White, A., Lee, J., et al. (2022). Reliably assessing the electronic structure of cytochrome P450 on today's classical computers and tomorrow's quantum computers. *Proc. Natl. Acad. Sci.*, **119**, e2203533119. doi:10.1073/pnas.2203533119. arXiv:2202.01244.

[438] Gondzio, J. (2012). Interior point methods 25 years later. *Eur. J. Oper. Res.*, **218**, 587–601. doi:10.1016/j.ejor.2011.09.017.

[439] Gong, W., Zhang, C., and Li, T. (2022). Robustness of quantum algorithms for nonconvex optimization. arXiv:2212.02548.

[440] Gonthier, J. F., Radin, M. D., Buda, C., et al. (2022). Measurements as a roadblock to near-term practical quantum advantage in chemistry: Resource analysis. *Phys. Rev. Res.*, **4**, 033154. doi:10.1103/PhysRevResearch.4.033154.

[441] Gonzalez-Conde, J., Rodríguez-Rozas, Á., Solano, E., and Sanz, M. (2021). Simulating option price dynamics with exponential quantum speedup. arXiv:2101.04023.

[442] Google AI Quantum, Arute, F., Arya, K., et al. (2020). Hartree–Fock on a superconducting qubit quantum computer. *Science*, **369**, 1084–1089. doi:10.1126/science.abb9811. arXiv:2004.04174.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[443] Gorman, D. J., Hemmerling, B., Megidish, E., et al. (2018). Engineering vibrationally assisted energy transfer in a trapped-ion quantum simulator. *Phys. Rev. X*, **8**, 011038. doi:10.1103/PhysRevX.8.011038. arXiv:1709.04064.

[444] Gottesman, D. (1996). Class of quantum error-correcting codes saturating the quantum Hamming bound. *Phys. Rev. A*, **54**, 1862–1868. doi:10.1103/PhysRevA.54.1862. arXiv:quant-ph/9604038.

[445] Gottesman, D. (1998). The Heisenberg representation of quantum computers. arXiv:quant-ph/9807006.

[446] Gottesman, D. (2010). An introduction to quantum error correction and fault-tolerant quantum computation. Pages 13–58 of: *Proceedings of Symposia in Applied Mathematics*, vol. 68. arXiv:0904.2557.

[447] Gottesman, D. (2014). Fault-tolerant quantum computation with constant overhead. *Quantum Inf. Comput.*, **14**, 1338–1372. doi:10.26421/QIC14.15-16-5. arXiv:1310.2984.

[448] Gottesman, D., and Chuang, I. L. (1999). Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations. *Nature*, **402**, 390–393. doi:10.1038/46503.

[449] Gourianov, N., Lubasch, M., Dolgov, S., et al. (2022). A quantum-inspired approach to exploit turbulence structures. *Nat. Comput. Sci.*, **2**, 30–37. doi:10.1038/s43588-021-00181-1. arXiv:2106.05782.

[450] Gouzien, E., Ruiz, D., Le Régent, F.-M., et al. (2023). Performance analysis of a repetition cat code architecture: Computing 256-bit elliptic curve logarithm in 9 hours with 126133 cat qubits. *Phys. Rev. Lett.*, **131**, 040602. doi:10.1103/PhysRevLett.131.040602. arXiv:2302.06639.

[451] Grant, E., Benedetti, M., Cao, S., et al. (2018). Hierarchical quantum classifiers. *npj Quant. Inf.*, **4**, 65. doi:10.1038/s41534-018-0116-9. arXiv:1804.03680.

[452] Grant, E., Humble, T. S., and Stump, B. (2021). Benchmarking quantum annealing controls with portfolio optimization. *Phys. Rev. Appl.*, **15**, 014012. doi:10.1103/PhysRevApplied.15.014012. arXiv:2007.03005.

[453] Grassl, M., Langenberg, B., Roetteler, M., and Steinwandt, R. (2016). Applying Grover's algorithm to AES: Quantum resource estimates. Pages 29–43 of: *PQCrypto*. doi:10.1007/978-3-319-29360-8_3. arXiv:1512.04965.

[454] Gray, J., and Chan, G. K.-L. (2024). Hyperoptimized approximate contraction of tensor networks with arbitrary geometry. *Phys. Rev. X*, **14**, 011009. doi:10.1103/PhysRevX.14.011009. arXiv:2206.07044.

[455] Gray, J., and Kourtis, S. (2021). Hyper-optimized tensor network contraction. *Quantum*, **5**, 410. doi:10.22331/q-2021-03-15-410. arXiv:2002.01935.

[456] Greenaway, S., Pol, W., and Sim, S. (2024). A case study against QSVT: Assessment of quantum phase estimation improved by signal processing techniques. arXiv:2404.01396.

[457] Griewank, A., and Walther, A. (2008). *Evaluating derivatives: Principles and techniques of algorithmic differentiation*. SIAM. doi:10.1137/1.9780898717761.

[458] Griffiths, R. B., and Niu, C.-S. (1996). Semiclassical Fourier transform for quantum computation. *Phys. Rev. Lett.*, **76**, 3228. doi:10.1103/PhysRevLett.76.3228. arXiv:quant-ph/9511007.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[459] Grigoriadis, M. D., and Khachiyan, L. G. (1995). A sublinear-time randomized approximation algorithm for matrix games. *Oper. Res. Lett.*, **18**, 53–58. doi:10.1016/0167-6377(95)00032-0.

[460] Grinko, D., Gacon, J., Zoufal, C., and Woerner, S. (2021). Iterative quantum amplitude estimation. *npj Quant. Inf.*, **7**, 52. doi:10.1038/s41534-021-00379-1. arXiv:1912.05559.

[461] Gross, C., and Bloch, I. (2017). Quantum simulations with ultracold atoms in optical lattices. *Science*, **357**, 995–1001. doi:10.1126/science.aal3837.

[462] Gross, D., Liu, Y.-K., Flammia, S. T., et al. (2010). Quantum state tomography via compressed sensing. *Phys. Rev. Lett.*, **105**, 150401. doi:10.1103/PhysRevLett.105.150401. arXiv:0909.3304.

[463] Grover, L., and Rudolph, T. (2002). Creating superpositions that correspond to efficiently integrable probability distributions. arXiv:quant-ph/0208112.

[464] Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. Pages 212–219 of: *STOC*. doi:10.1145/237814.237866. arXiv:quant-ph/9605043.

[465] Grover, L. K. (2000). Synthesis of quantum superpositions by quantum computation. *Phys. Rev. Lett.*, **85**, 1334. doi:10.1103/PhysRevLett.85.1334.

[466] Grover, L. K. (2005). Fixed-point quantum search. *Phys. Rev. Lett.*, **95**, 150501. doi:10.1103/PhysRevLett.95.150501. arXiv:quant-ph/0503205.

[467] Gu, S., Retzker, A., and Kubica, A. (2023). Fault-tolerant quantum architectures based on erasure qubits. arXiv:2312.14060.

[468] Gui, K., Dalzell, A. M., Achille, A., et al. (2024). Spacetime-efficient low-depth quantum state preparation with applications. *Quantum*, **8**, 1257. doi:10.22331/q-2024-02-15-1257. arXiv:2303.02131.

[469] Gunn, S., and Kornerup, N. (2019). Review of a quantum algorithm for Betti numbers. arXiv:1906.07673.

[470] Gurobi Optimization. (2024). *Gurobi optimizer reference manual*. Version 11.0, accessed: 2024-05-09.

[471] Guta, M., Kahn, J., Kueng, R., and Tropp, J. A. (2020). Fast state tomography with optimal error bounds. *J. Phys. A*, **53**, 204001. doi:10.1088/1751-8121/ab8111. arXiv:1809.11162.

[472] Gysbers, P., Hagen, G., Holt, J. D., et al. (2019). Discrepancy between experimental and theoretical β-decay rates resolved from first principles. *Nat. Phys.*, **15**, 428–431. doi:10.1038/s41567-019-0450-7. arXiv:1903.00047.

[473] Gyurik, C., and Dunjko, V. (2023). Exponential separations between classical and quantum learners. arXiv:2306.16028.

[474] Gyurik, C., Cade, C., and Dunjko, V. (2022). Towards quantum advantage via topological data analysis. *Quantum*, **6**, 855. doi:10.22331/q-2022-11-10-855.

[475] Gyurik, C., Dunjko, V., et al. (2023). Structural risk minimization for quantum linear classifiers. *Quantum*, **7**, 893. doi:10.22331/q-2023-01-13-893. arXiv:2105.05566.

[476] Ha, J., Lee, J., and Heo, J. (2022). Resource analysis of quantum computing with noisy qubits for Shor's factoring algorithms. *Quantum Inf. Process.*, **21**, 60. doi:10.1007/s11128-021-03398-1.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[477] Haah, J. (2019). Product decomposition of periodic functions in quantum signal processing. *Quantum*, **3**, 190. doi:10.22331/q-2019-10-07-190. arXiv:1806.10236.

[478] Haah, J., Harrow, A. W., Ji, Z., et al. (2017). Sample-optimal tomography of quantum states. *IEEE Trans. Inf. Theory*, **63**, 5628–5641. doi:10.1109/TIT.2017.2719044. arXiv:1508.01797.

[479] Haah, J., Hastings, M. B., Kothari, R., and Low, G. H. (2018). Quantum algorithm for simulating real time evolution of lattice Hamiltonians. Pages 350–360 of: *FOCS*. doi:10.1109/FOCS.2018.00041. arXiv:1801.03922.

[480] Haah, J., Kothari, R., O'Donnell, R., and Tang, E. (2023). Query-optimal estimation of unitary channels in diamond distance. arXiv:2302.14066.

[481] Hackbusch, W. (2016). *Iterative solution of large sparse systems of equations*. 2nd edn. Vol. 95. Springer. doi:10.1007/978-3-319-28483-5.

[482] Hadfield, S., Wang, Z., O'Gorman, B., et al. (2019). From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, **12**, 34. doi:10.3390/a12020034. arXiv:1709.03489.

[483] Haferkamp, J., Hangleiter, D., Eisert, J., and Gluza, M. (2020). Contracting projected entangled pair states is average-case hard. *Phys. Rev. Res.*, **2**, 013010. doi:10.1103/PhysRevResearch.2.013010. arXiv:1810.00738.

[484] Hagan, M., and Wiebe, N. (2022). Composite quantum simulations. arXiv:2206.06409.

[485] Hagen, G., Papenbrock, T., Hjorth-Jensen, M., and Dean, D. J. (2014). Coupled-cluster computations of atomic nuclei. *Rep. Prog. Phys.*, **77**, 096302. doi:10.1088/0034-4885/77/9/096302. arXiv:1312.7872.

[486] Hales, L., and Hallgren, S. (2000). An improved quantum Fourier transform algorithm and applications. Pages 515–525 of: *FOCS*. doi:10.1109/SFCS.2000.892139.

[487] Halimeh, J. C., Lang, H., Mildenberger, J., et al. (2021). Gauge-symmetry protection using single-body terms. *PRX Quantum*, **2**, 040311. doi:10.1103/PRXQuantum.2.040311. arXiv:2007.00668.

[488] Halkier, A., Helgaker, T., Jørgensen, P., et al. (1998). Basis-set convergence in correlated calculations on Ne, N_2, and H_2O. *Chem. Phys. Lett.*, **286**, 243–252. doi:10.1016/S0009-2614(98)00111-0.

[489] Hammer, H.-W., König, S., and van Kolck, U. (2020). Nuclear effective field theory: Status and perspectives. *Rev. Mod. Phys.*, **92**, 025004. doi:10.1103/RevModPhys.92.025004.

[490] Hamoudi, Y., and Magniez, F. (2019). Quantum Chebyshev's inequality and applications. Pages 69:1–69:16 of: *ICALP*. doi:10.4230/LIPIcs.ICALP.2019.69. arXiv:1807.06456.

[491] Han, J. Y., and Rebentrost, P. (2022). Quantum advantage for multi-option portfolio pricing and valuation adjustments. arXiv:2203.04924.

[492] Häner, T., and Steiger, D. S. (2017). 0.5 petabyte simulation of a 45-qubit quantum circuit. In: *SC*. doi:10.1145/3126908.3126947. arXiv:1704.01127.

[493] Häner, T., Roetteler, M., and Svore, K. M. (2017). Factoring using 2n + 2 qubits with Toffoli based modular multiplication. *Quantum Inf. Comput.*, **17**, 673–684. doi:10.26421/QIC17.7-8-7. arXiv:1611.07995.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[494] Häner, T., Roetteler, M., and Svore, K. M. (2018). Optimizing quantum circuits for arithmetic. arXiv:1805.12445.

[495] Häner, T., Jaques, S., Naehrig, M., et al. (2020). Improved quantum circuits for elliptic curve discrete logarithms. Pages 425–444 of: *PQCrypto*. doi:10.1007/978-3-030-44223-1_23. arXiv:2001.09580.

[496] Hann, C. T., Lee, G., Girvin, S., and Jiang, L. (2021). Resilience of quantum random access memory to generic noise. *PRX Quantum*, **2**, 020311. doi:10.1103/PRXQuantum.2.020311. arXiv:2012.05340.

[497] Hansen, T. D., Kaplan, H., Zamir, O., and Zwick, U. (2019). Faster k-SAT algorithms using biased-PPSZ. Pages 578–589 of: *STOC*. doi:10.1145/3313276.3316359.

[498] Harrow, A. W. (2020). Small quantum computers and large classical data sets. arXiv:2004.00026.

[499] Harrow, A. W., and Wei, A. Y. (2020). Adaptive quantum simulated annealing for Bayesian inference and estimating partition functions. Pages 193–212 of: *SODA*. doi:10.1137/1.9781611975994.12. arXiv:1907.09965.

[500] Harrow, A. W., Hassidim, A., and Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Phys. Rev. Lett.*, **103**, 150502. doi:10.1103/PhysRevLett.103.150502. arXiv:0811.3171.

[501] Hart, O., Gopalakrishnan, S., and Castelnovo, C. (2021). Logarithmic entanglement growth from disorder-free localization in the two-leg compass ladder. *Phys. Rev. Lett.*, **126**, 227202. doi:10.1103/PhysRevLett.126.227202. arXiv:2009.00618.

[502] Harvey, D., and van der Hoeven, J. (2021). Integer multiplication in time $o(n \log n)$. *Ann. Math.*, **193**, 563 – 617. doi:10.4007/annals.2021.193.2.4.

[503] Hastings, M. (2010). *Theoretical computer science stack exchange: The complexity of area-lawed Hamiltonians*. https://cstheory.stackexchange.com/questions/3083/the-complexity-of-area-lawed-hamiltonians, accessed: 2024-10-29.

[504] Hastings, M. B. (2007). An area law for one-dimensional quantum systems. *J. Stat. Mech. Theory Exp.*, **2007**, P08024. doi:10.1088/1742-5468/2007/08/P08024. arXiv:0705.2024.

[505] Hastings, M. B. (2018). A short path quantum algorithm for exact optimization. *Quantum*, **2**, 78. doi:10.22331/q-2018-07-26-78. arXiv:1802.10124.

[506] Hastings, M. B. (2018). Weaker assumptions for the short path optimization algorithm. doi:10.48550/arXiv.1807.03758. arXiv:1807.03758.

[507] Hastings, M. B. (2019). Classical and quantum bounded depth approximation algorithms. arXiv:1905.07047.

[508] Hastings, M. B. (2019). The short path algorithm applied to a toy model. *Quantum*, **3**, 145. doi:10.22331/q-2019-05-20-145. arXiv:1901.03884.

[509] Hastings, M. B. (2020). Classical and quantum algorithms for tensor principal component analysis. *Quantum*, **4**, 237. doi:10.22331/q-2020-02-27-237. arXiv:1907.12724.

[510] Hastings, M. B. (2020). The power of adiabatic quantum computation with no sign problem. arXiv:2005.03791.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[511] Hastings, M. B., and O'Donnell, R. (2022). Optimizing strongly interacting fermionic Hamiltonians. Pages 776–789 of: *STOC*. doi:10.1145/3519935.3519960. arXiv:2110.10701.

[512] Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, **57**, 97–109. doi:10.1093/biomet/57.1.97.

[513] Havlíček, V., Córcoles, A. D., Temme, K., et al. (2019). Supervised learning with quantum-enhanced feature spaces. *Nature*, **567**, 209–212. doi:10.1038/s41586-019-0980-2. arXiv:1804.11326.

[514] Hayakawa, R. (2022). Quantum algorithm for persistent Betti numbers and topological data analysis. *Quantum*, **6**, 873. doi:10.22331/q-2022-12-07-873. arXiv:2111.00433.

[515] Heinrich, S. (2002). Quantum summation with an application to integration. *J. Complex.*, **18**, 1–50. doi:10.1006/jcom.2001.0629. arXiv:quant-ph/0105116.

[516] Hejazi, K., Zini, M. S., and Arrazola, J. M. (2024). Better bounds for low-energy product formulas. arXiv:2402.10362.

[517] Hen, I., and Young, A. P. (2011). Exponential complexity of the quantum adiabatic algorithm for certain satisfiability problems. *Phys. Rev. E*, **84**, 061152. doi:10.1103/PhysRevE.84.061152. arXiv:1109.6872.

[518] Henderson, J. M., Podzorova, M., Cerezo, M., et al. (2023). Quantum algorithms for geologic fracture networks. *Sci. Rep.*, **13**, 2906. doi:10.1038/s41598-023-29643-4. arXiv:2210.11685.

[519] Hensel, F., Moor, M., and Rieck, B. (2021). A survey of topological machine learning methods. *Front. Artif. Intell.*, **4**, 681108. doi:10.3389/frai.2021.681108.

[520] Henstra, F. (2023). *The multiplicative weights method in quantum computing*. Master's thesis, Utrecht University.

[521] Herbert, S. (2021). No quantum speedup with Grover–Rudolph state preparation for quantum Monte Carlo integration. *Phys. Rev. E*, **103**, 063302. doi:10.1103/PhysRevE.103.063302. arXiv:2101.02240.

[522] Herbert, S. (2022). Quantum Monte Carlo integration: The full advantage in minimal circuit depth. *Quantum*, **6**, 823. doi:10.22331/q-2022-09-29-823. arXiv:2105.09100.

[523] Hergert, H. (2020). A guided tour of ab initio nuclear many-body theory. *Front. Phys.*, **8**. doi:10.3389/fphy.2020.00379. arXiv:2008.05061.

[524] Herman, D., Shaydulin, R., Sun, Y., et al. (2022). Portfolio optimization via quantum Zeno dynamics on a quantum processor. arXiv:2209.15024.

[525] Herman, D., Googin, C., Liu, X., et al. (2023). Quantum computing for finance. *Nat. Rev. Phys.*, **5**, 450–465. doi:10.1038/s42254-023-00603-1. arXiv:2201.02773.

[526] Hertli, T. (2014). 3-SAT faster and simpler—Unique-SAT bounds for PPSZ hold in general. *SIAM J. Comp.*, **43**, 718–729. doi:10.1137/120868177. arXiv:1103.2165.

[527] Hestenes, M. R., and Stiefel, E. (1952). Methods of conjugate gradients for solving linear systems. *J. Res. Natl. Bur. Stand.*, **49**, 409–435.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[528] Higgott, O. (2022). Pymatching: A Python package for decoding quantum codes with minimum-weight perfect matching. *ACM Trans. Quantum Comput.*, **3**. doi:10.1145/3505637. arXiv:2105.13082.

[529] Higgott, O., and Gidney, C. (2023). Sparse blossom: Correcting a million errors per core second with minimum-weight matching. arXiv:2303.15933.

[530] Higgott, O., Bohdanowicz, T. C., Kubica, A., et al. (2023). Improved decoding of circuit noise and fragile boundaries of tailored surface codes. *Phys. Rev. X*, **13**, 031007. doi:10.1103/PhysRevX.13.031007. arXiv:2203.04948.

[531] Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Comput.*, **14**, 1771–1800. doi:10.1162/089976602760128018.

[532] Hoefler, T., Häner, T., and Troyer, M. (2023). Disentangling hype from practicality: On realistically achieving quantum advantage. *Commun. ACM*, **66**, 82–87. doi:10.1145/3571725.

[533] Hogben, H., Krzystyniak, M., Charnock, G., et al. (2011). Spinach—a software library for simulation of spin dynamics in large spin systems. *J. Magn. Reson.*, **208**, 179–194. doi:10.1016/j.jmr.2010.11.008.

[534] Holmes, Z., Sharma, K., Cerezo, M., and Coles, P. J. (2022). Connecting ansatz expressibility to gradient magnitudes and barren plateaus. *PRX Quantum*, **3**, 010313. doi:10.1103/PRXQuantum.3.010313. arXiv:2101.02138.

[535] Horsman, D., Fowler, A. G., Devitt, S., and Meter, R. V. (2012). Surface code quantum computing by lattice surgery. *New J. Phys.*, **14**, 123011. doi:10.1088/1367-2630/14/12/123011. arXiv:1111.4022.

[536] Hoyle, D., and Rattray, M. (2003). PCA learning for sparse high-dimensional data. *Europhys. Lett.*, **62**, 117. doi:10.1209/epl/i2003-00370-1.

[537] Huang, B., Jiang, S., Song, Z., et al. (2022). A faster quantum algorithm for semidefinite programming via robust IPM framework. arXiv:2207.11154.

[538] Huang, B., Jiang, S., Song, Z., et al. (2022). Solving SDP faster: A robust IPM framework and efficient implementation. Pages 233–244 of: *FOCS*. doi:10.1109/FOCS54457.2022.00029. arXiv:2101.08208.

[539] Huang, B., Chen, Y.-T., Gupt, B., et al. (2024). Evaluating a quantum-classical quantum Monte Carlo algorithm with matchgate shadows. arXiv:2404.18303.

[540] Huang, H.-Y., Kueng, R., and Preskill, J. (2020). Predicting many properties of a quantum system from very few measurements. *Nat. Phys.*, **16**, 1050–1057. doi:10.1038/s41567-020-0932-7. arXiv:2002.08953.

[541] Huang, H.-Y., Bharti, K., and Rebentrost, P. (2021). Near-term quantum algorithms for linear systems of equations with regression loss functions. *New J. Phys.*, **23**, 113021. doi:10.1088/1367-2630/ac325f. arXiv:1909.07344.

[542] Huang, H.-Y., Broughton, M., Mohseni, M., et al. (2021). Power of data in quantum machine learning. *Nat. Commun.*, **12**, 2631. doi:10.1038/s41467-021-22539-9. arXiv:2011.01938.

[543] Huang, H.-Y., Broughton, M., Cotler, J., et al. (2022). Quantum advantage in learning from experiments. *Science*, **376**, 1182–1186. doi:10.1126/science.abn7293. arXiv:2112.00778.

[544] Huang, S., and Brown, K. R. (2021). Between Shor and Steane: A unifying construction for measuring error syndromes. *Phys. Rev. Lett.*, **127**. doi:10.1103/physrevlett.127.090505. arXiv:2012.15403.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[545] Huang, Y. (2021). Two-dimensional local Hamiltonian problem with area laws is QMA-complete. *J. Comput. Phys.*, **443**, 110534. doi:10.1016/j.jcp.2021.110534. Earlier version in *ISIT'20*. arXiv:1411.6614.

[546] Hubbard, J., and Flowers, B. H. (1963). Electron correlations in narrow energy bands. *Proc. R. Soc. A*, **276**, 238–257. doi:10.1098/rspa.1963.0204.

[547] Hubregtsen, T., Wierichs, D., Gil-Fuster, E., et al. (2022). Training quantum embedding kernels on near-term quantum computers. *Phys. Rev. A*, **106**, 042431. doi:10.1103/PhysRevA.106.042431. arXiv:2105.02276.

[548] Huggins, W., Patil, P., Mitchell, B., et al. (2019). Towards quantum machine learning with tensor networks. *Quantum Sci. Technol.*, **4**, 024001. doi:10.1088/2058-9565/aaea94. arXiv:1803.11537.

[549] Huggins, W. J., Wan, K., McClean, J., et al. (2022). Nearly optimal quantum algorithm for estimating multiple expectation values. *Phys. Rev. Lett.*, **129**, 240501. doi:10.1103/PhysRevLett.129.240501. arXiv:2111.09283.

[550] Huggins, W. J., O'Gorman, B. A., Rubin, N. C., et al. (2022). Unbiasing fermionic quantum Monte Carlo with a quantum computer. *Nature*, **603**, 416–420. doi:10.1038/s41586-021-04351-z. arXiv:2106.16235.

[551] Huggins, W. J., Leimkuhler, O., Stetina, T. F., and Whaley, K. B. (2024). Efficient state preparation for the quantum simulation of molecules in first quantization. arXiv:2407.00249.

[552] Huh, J. (2011). *Unified description of vibronic transitions with coherent states*. PhD thesis, Johann Wolfgang Goethe-Universität.

[553] Huh, J., Guerreschi, G. G., Peropadre, B., et al. (2015). Boson sampling for molecular vibronic spectra. *Nat. Photonics*, **9**, 615–620. doi:10.1038/nphoton.2015.153. arXiv:1412.8427.

[554] Hull, J. (2017). *Options, futures, and other derivatives*. Pearson.

[555] Hunter-Jones, N. R. (2018). *Chaos and randomness in strongly-interacting quantum systems*. PhD thesis, California Institute of Technology. doi:10.7907/BHZ5-HV76.

[556] Huyghebaert, J., and Raedt, H. D. (1990). Product formula methods for time-dependent Schrodinger problems. *J. Phys. A*, **23**, 5777. doi:10.1088/0305-4470/23/24/019.

[557] Iijima, N., Imamura, S., Morita, M., et al. (2023). Towards accurate quantum chemical calculations on noisy quantum computers. arXiv:2311.09634.

[558] Impagliazzo, R., Paturi, R., and Zane, F. (2001). Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, **63**, 512–530. doi:10.1006/jcss.2001.1774.

[559] Information Technology Laboratory. (2001). *Advanced Encryption Standard (AES)*. Tech. rept. FIPS 197. National Institute of Standards and Technology, Gaithersburg, MD. doi:10.6028/NIST.FIPS.197-upd1.

[560] Itani, W., Sreenivasan, K. R., and Succi, S. (2023). Quantum algorithm for lattice Boltzmann (QALB) simulation of incompressible fluids with a nonlinear collision term. arXiv:2304.05915.

[561] Ivanov, A. V., Sünderhauf, C., Holzmann, N., et al. (2023). Quantum computation for periodic solids in second quantization. *Phys. Rev. Res.*, **5**, 013200. doi:10.1103/PhysRevResearch.5.013200. arXiv:2210.02403.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[562] Ivanov, A. V., Patterson, A., Bothe, M., et al. (2024). Quantum computation of electronic structure with projector augmented-wave method and plane wave basis set. arXiv:2408.03159.

[563] Iyer, P., and Poulin, D. (2015). Hardness of decoding quantum stabilizer codes. *IEEE Trans. Inf. Theory*, **61**, 5209–5223. doi:10.1109/TIT.2015.2422294. arXiv:1310.3235.

[564] Iyer, P., and Poulin, D. (2018). A small quantum computer is needed to optimize fault-tolerant protocols. *Quantum Sci. Technol.*, **3**, 030504. doi:10.1088/2058-9565/aab73c. arXiv:1711.04736.

[565] Jackson, A., Kapourniotis, T., and Datta, A. (2023). Partition-function estimation: Quantum and quantum-inspired algorithms. *Phys. Rev. A*, **107**, 012421. doi:10.1103/PhysRevA.107.012421. arXiv:2208.00930.

[566] Jang, K., Baksi, A., Kim, H., et al. (2024). Quantum analysis of AES. Pages 51–90 of: *Implementation and Analysis of Ciphers in Quantum Computing*. doi:10.1007/978-981-97-0025-7. ePrint:2022/683.

[567] Jansen, S., Ruskai, M.-B., and Seiler, R. (2007). Bounds for the adiabatic approximation with applications to quantum computation. *J. Math. Phys.*, **48**, 102111. doi:10.1063/1.2798382. arXiv:quant-ph/0603175.

[568] Jaques, S., and Rattew, A. G. (2023). QRAM: A survey and critique. arXiv:2305.10310.

[569] Jarret, M., and Wan, K. (2018). Improved quantum backtracking algorithms using effective resistance estimates. *Phys. Rev. A*, **97**, 022337. doi:10.1103/PhysRevA.97.022337. arXiv:1711.05295.

[570] Jarret, M., Jordan, S. P., and Lackey, B. (2016). Adiabatic optimization versus diffusion Monte Carlo methods. *Phys. Rev. A*, **94**, 042318. doi:10.1103/PhysRevA.94.042318. arXiv:1607.03389.

[571] Jennings, D., Lostaglio, M., Lowrie, R. B., et al. (2023). The cost of solving linear differential equations on a quantum computer: Fast-forwarding to explicit resource counts. arXiv:2309.07881.

[572] Jennings, D., Lostaglio, M., Pallister, S., et al. (2023). Efficient quantum linear solver algorithm with detailed running costs. arXiv:2305.11352.

[573] Jerbi, S., Gyurik, C., Marshall, S., et al. (2021). Parametrized quantum policies for reinforcement learning. Pages 28362–28375 of: *NeurIPS*. arXiv:2103.05577.

[574] Jiang, H., Kathuria, T., Lee, Y. T., et al. (2020). A faster interior point method for semidefinite programming. Pages 910–918 of: *FOCS*. doi:10.1109/FOCS46700.2020.00089. arXiv:2009.10217.

[575] Jiang, H., Lee, Y. T., Song, Z., and Wong, S. C.-W. (2020). An improved cutting plane method for convex optimization, convex-concave games, and its applications. Pages 944–953 of: *STOC*. doi:10.1145/3357713.3384284. arXiv:2004.04250.

[576] Jiang, Z., Sung, K. J., Kechedzhi, K., et al. (2018). Quantum algorithms to simulate many-body physics of correlated fermions. *Phys. Rev. Appl.*, **9**, 44036. doi:10.1103/PhysRevApplied.9.044036. arXiv:1711.05395.

[577] Jin, C., Netrapalli, P., and Jordan, M. I. (2018). Accelerated gradient descent escapes saddle points faster than gradient descent. Pages 1042–1085 of: *COLT*, vol. 75. arXiv:1711.10456.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[578] Jin, S., and Liu, N. (2022). Quantum algorithms for computing observables of nonlinear partial differential equations. arXiv:2202.07834.

[579] Jin, S., Liu, N., and Yu, Y. (2022). Quantum simulation of partial differential equations via Schrodingerisation. arXiv:2212.13969.

[580] Jin, S., Liu, N., and Yu, Y. (2022). Time complexity analysis of quantum difference methods for linear high dimensional and multiscale partial differential equations. *J. Comput. Phys.*, **471**, 111641. doi:10.1016/j.jcp.2022.111641. arXiv:2202.04537.

[581] Jin, S., Liu, N., and Yu, Y. (2023). Quantum simulation of partial differential equations: Applications and detailed analysis. *Phys. Rev. A*, **108**, 032603. doi:10.1103/PhysRevA.108.032603. arXiv:2212.14703.

[582] Jochym-O'Connor, T., and Bartlett, S. D. (2016). Stacked codes: Universal fault-tolerant quantum computation in a two-dimensional layout. *Phys. Rev. A*, **93**, 022323. doi:10.1103/PhysRevA.93.022323. arXiv:1509.04255.

[583] Jochym-O'Connor, T., Kubica, A., and Yoder, T. J. (2018). Disjointness of stabilizer codes and limitations on fault-tolerant logical gates. *Phys. Rev. X*, **8**, 021047. doi:10.1103/physrevx.8.021047. arXiv:1710.07256.

[584] Jóczik, S., Zimborás, Z., Majoros, T., and Kiss, A. (2022). A cost-efficient approach towards computational fluid dynamics simulations on quantum devices. *Appl. Sci.*, **12**, 2873. doi:10.3390/app12062873.

[585] Joó, B., Jung, C., Christ, N. H., et al. (2019). Status and future perspectives for lattice gauge theory calculations to the exascale and beyond. *Eur. Phys. J. A*, **55**, 199. doi:10.1140/epja/i2019-12919-7. arXiv:1904.09725.

[586] Jordan, S. (2011). *Quantum algorithm zoo*. https://quantumalgorithmzoo.org/, accessed: 2024-10-29.

[587] Jordan, S. P. (2005). Fast quantum algorithm for numerical gradient estimation. *Phys. Rev. Lett.*, **95**, 050501. doi:10.1103/PhysRevLett.95.050501. arXiv:quant-ph/0405146.

[588] Jordan, S. P., Lee, K. S. M., and Preskill, J. (2012). Quantum algorithms for quantum field theories. *Science*, **336**, 1130–1133. doi:10.1126/science.1217069. arXiv:1111.3633.

[589] Jordan, S. P., Krovi, H., Lee, K. S. M., and Preskill, J. (2018). BQP-completeness of scattering in scalar quantum field theory. *Quantum*, **2**, 44. doi:10.22331/q-2018-01-08-44. arXiv:1703.00454.

[590] Joseph, I. (2020). Koopman–von Neumann approach to quantum simulation of nonlinear classical dynamics. *Phys. Rev. Res.*, **2**, 043102. doi:10.1103/PhysRevResearch.2.043102. arXiv:2003.09980.

[591] Kadowaki, T., and Nishimori, H. (1998). Quantum annealing in the transverse Ising model. *Phys. Rev. E*, **58**, 5355–5363. doi:10.1103/PhysRevE.58.5355. arXiv:cond-mat/9804280.

[592] Kahanamoku-Meyer, G. D., and Yao, N. Y. (2024). Fast quantum integer multiplication with zero ancillas. arXiv:2403.18006.

[593] Kan, A., and Nam, Y. (2021). Lattice quantum chromodynamics and electrodynamics on a universal quantum computer. arXiv:2107.12769.

[594] Kanagawa, M., Hennig, P., Sejdinovic, D., and Sriperumbudur, B. K. (2018). Gaussian processes and kernel methods: A review on connections and equivalences. arXiv:1807.02582.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[595] Kandala, A., Mezzacapo, A., Temme, K., et al. (2017). Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. *Nature*, **549**, 242–246. doi:10.1038/nature23879. arXiv:1704.05018.

[596] Kaneko, K., Miyamoto, K., Takeda, N., and Yoshino, K. (2022). Quantum pricing with a smile: Implementation of local volatility model on quantum computer. *EPJ Quantum Technol.*, **9**, 7. doi:10.1140/epjqt/s40507-022-00125-2. arXiv:2007.01467.

[597] Kang, M., Nuomin, H., Chowdhury, S. N., et al. (2024). Seeking a quantum advantage with trapped-ion quantum simulations of condensed-phase chemical dynamics. *Nat. Rev. Chem.*, **8**, 340–358. doi:10.1038/s41570-024-00595-1. arXiv:2305.03156.

[598] Kaplan, D. B. (2011). Chiral symmetry and lattice fermions. Pages 223–272 of: *Modern perspectives in lattice QCD: Quantum field theory and high performance computing: Lecture notes of the Les Houches Summer School: Volume 93, August 2009*. doi:10.1093/acprof:oso/9780199691609.003.0004. arXiv:0912.2560.

[599] Kaplan, M., Leurent, G., Leverrier, A., and Naya-Plasencia, M. (2016). Breaking symmetric cryptosystems using quantum period finding. Pages 207–237 of: *CRYPTO*. doi:10.1007/978-3-662-53008-5_8. arXiv:1602.05973.

[600] Karamlou, A. H., Simon, W. A., Katabarwa, A., et al. (2021). Analyzing the performance of variational quantum factoring on a superconducting quantum processor. *npj Quant. Inf.*, **7**, 156. doi:10.1038/s41534-021-00478-z. arXiv:2012.07825.

[601] Kassal, I., Jordan, S. P., Love, P. J., et al. (2008). Polynomial-time quantum algorithm for the simulation of chemical dynamics. *Proc. Natl. Acad. Sci.*, **105**, 18681–18686. doi:10.1073/pnas.0808245105. arXiv:0801.2986.

[602] Kastoryano, M., and Pancotti, N. (2022). A highly efficient tensor network algorithm for multi-asset Fourier options pricing. arXiv:2203.02804.

[603] Kastoryano, M. J., and Brandão, F. G. S. L. (2016). Quantum Gibbs samplers: The commuting case. *Commun. Math. Phys.*, **344**, 915–957. doi:10.1007/s00220-016-2641-8. arXiv:1409.3435.

[604] Kattemölle, J., and van Wezel, J. (2022). Variational quantum eigensolver for the Heisenberg antiferromagnet on the kagome lattice. *Phys. Rev. B*, **106**, 214429. doi:10.1103/PhysRevB.106.214429.

[605] Katz, J., and Lindell, Y. (2021). *Introduction to modern cryptography: Third edition*. CRC Press.

[606] Kaye, P., Laflamme, R., and Mosca, M. (2006). *An introduction to quantum computing*. Oxford University Press. doi:10.1093/oso/9780198570004.001.0001.

[607] Kempe, J., Kitaev, A., and Regev, O. (2006). The complexity of the local Hamiltonian problem. *SIAM J. Comp.*, **35**, 1070–1097. doi:10.1137/S0097539704445226. Earlier version in *FSTTCS'04*. arXiv:quant-ph/0406180.

[608] Kerenidis, I., and Prakash, A. (2017). Quantum recommendation systems. Pages 49:1–49:21 of: *ITCS*. doi:10.4230/LIPIcs.ITCS.2017.49. arXiv:1603.08675.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[609] Kerenidis, I., and Prakash, A. (2020). Quantum gradient descent for linear systems and least squares. *Phys. Rev. A*, **101**, 022316. doi:10.1103/PhysRevA.101.022316. arXiv:1704.04992.

[610] Kerenidis, I., and Prakash, A. (2020). A quantum interior point method for LPs and SDPs. *ACM Trans. Quantum Comput.*, **1**. doi:10.1145/3406306. arXiv:1808.09266.

[611] Kerenidis, I., Prakash, A., and Szilágyi, D. (2019). Quantum algorithms for portfolio optimization. Pages 147–155 of: *AFT*. doi:10.1145/3318041.3355465. arXiv:1908.08040.

[612] Kerenidis, I., Prakash, A., and Szilágyi, D. (2021). Quantum algorithms for second-order cone programming and support vector machines. *Quantum*, **5**, 427. doi:10.22331/q-2021-04-08-427. arXiv:1908.06720.

[613] Khatri, S., LaRose, R., Poremba, A., et al. (2019). Quantum-assisted quantum compiling. *Quantum*, **3**, 140. doi:10.22331/q-2019-05-13-140. arXiv:1807.00800.

[614] Kieferová, M., and Wiebe, N. (2017). Tomography and generative training with quantum Boltzmann machines. *Phys. Rev. A*, **96**, 062327. doi:10.1103/PhysRevA.96.062327. arXiv:1612.05204.

[615] Kieferová, M., Scherer, A., and Berry, D. W. (2019). Simulating the dynamics of time-dependent Hamiltonians with a truncated Dyson series. *Phys. Rev. A*, **99**, 042314. doi:10.1103/PhysRevA.99.042314. arXiv:1805.00582.

[616] Kim, I. H., and Swingle, B. (2017). Robust entanglement renormalization on a noisy quantum computer. arXiv:1711.07500.

[617] Kim, I. H., Liu, Y.-H., Pallister, S., et al. (2022). Fault-tolerant resource estimate for quantum chemical simulations: Case study on Li-ion battery electrolyte molecules. *Phys. Rev. Res.*, **4**, 023019. doi:10.1103/PhysRevResearch.4.023019. arXiv:2104.10653.

[618] Kim, S. K., McMahon, P. L., and Olukotun, K. (2010). A large-scale architecture for restricted Boltzmann machines. Pages 201–208 of: *FCCM*. doi:10.1109/FCCM.2010.38.

[619] Kimmel, S., Lin, C. Y.-Y., Low, G. H., et al. (2017). Hamiltonian simulation with optimal sample complexity. *npj Quant. Inf.*, **3**, 13. doi:10.1038/s41534-017-0013-7. arXiv:1608.00281.

[620] King, R., and Kohler, T. (2023). Promise clique homology on weighted graphs is QMA_1-hard and contained in QMA. arXiv:2311.17234.

[621] Kiss, O., Grossi, M., and Roggero, A. (2023). Importance sampling for stochastic quantum simulations. *Quantum*, **7**, 977. doi:10.22331/q-2023-04-13-977. arXiv:2212.05952.

[622] Kitaev, A. (2015). *A simple model of quantum holography*. Video of talk: https://www.youtube.com/watch?v=wFH1huu9Jcs (part 1), https://www.youtube.com/watch?v=q8amZLOElB0 (part 2), accessed: 2023-09-30.

[623] Kitaev, A., and Webb, W. A. (2008). Wavefunction preparation and resampling using a quantum computer. arXiv:0801.0342.

[624] Kitaev, A. Y. (1995). Quantum measurements and the abelian stabilizer problem. arXiv:quant-ph/9511026.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[625] Kitaev, A. Y. (1997). Quantum computations: Algorithms and error correction. *Russ. Math. Surv.*, **52**, 1191. doi:10.1070/RM1997v052n06ABEH002155.

[626] Kitaev, A. Y. (2003). Fault-tolerant quantum computation by anyons. *Ann. Phys.*, **303**, 2–30. doi:10.1016/S0003-4916(02)00018-0. arXiv:quant-ph/9707021.

[627] Kitaev, A. Y., Shen, A., Vyalyi, M. N., and Vyalyi, M. N. (2002). *Classical and quantum computation*. American Mathematical Society.

[628] Kivlichan, I. D., Wiebe, N., Babbush, R., and Aspuru-Guzik, A. (2017). Bounding the costs of quantum simulation of many-body physics in real space. *J. Phys. A*, **50**, 305301. doi:10.1088/1751-8121/aa77b8. arXiv:1608.05696.

[629] Kivlichan, I. D., McClean, J., Wiebe, N., et al. (2018). Quantum simulation of electronic structure with linear depth and connectivity. *Phys. Rev. Lett.*, **120**, 110501. doi:10.1103/PhysRevLett.120.110501. arXiv:1711.04789.

[630] Kivlichan, I. D., Gidney, C., Berry, D. W., et al. (2020). Improved fault-tolerant quantum simulation of condensed-phase correlated electrons via Trotterization. *Quantum*, **4**, 296. doi:10.22331/q-2020-07-16-296. arXiv:1902.10673.

[631] Klco, N., Dumitrescu, E. F., McCaskey, A. J., et al. (2018). Quantum-classical computation of Schwinger model dynamics using quantum computers. *Phys. Rev. A*, **98**, 032331. doi:10.1103/PhysRevA.98.032331. arXiv:1803.03326.

[632] Klco, N., Roggero, A., and Savage, M. J. (2022). Standard model physics and the digital quantum revolution: Thoughts about the interface. *Rep. Prog. Phys.*, **85**, 064301. doi:10.1088/1361-6633/ac58a4. arXiv:2107.04769.

[633] Knill, E. (2004). Fault-tolerant postselected quantum computation: Schemes. arXiv:quant-ph/0402171.

[634] Knill, E. (2005). Quantum computing with realistically noisy devices. *Nature*, **434**, 39–44. doi:10.1038/nature03350. arXiv:quant-ph/0410199.

[635] Knill, E., and Laflamme, R. (1998). Power of one bit of quantum information. *Phys. Rev. Lett.*, **81**, 5672–5675. doi:10.1103/PhysRevLett.81.5672. arXiv:quant-ph/9802037.

[636] Knill, E., Laflamme, R., and Zurek, W. H. (1998). Resilient quantum computation: error models and thresholds. *Proc. R. Soc. A*, **454**, 365–384. doi:10.1098/rspa.1998.0166. arXiv:quant-ph/9702058.

[637] Knill, E., Ortiz, G., and Somma, R. D. (2007). Optimal quantum measurements of expectation values of observables. *Phys. Rev. A*, **75**, 012328. doi:10.1103/PhysRevA.75.012328. arXiv:quant-ph/0607019.

[638] Knysh, S., and Smelyanskiy, V. (2010). On the relevance of avoided crossings away from quantum critical point to the complexity of quantum adiabatic algorithm. doi:10.48550/arXiv.1005.3011. arXiv:1005.3011.

[639] Koblitz, N. (1987). Elliptic curve cryptosystems. *Math. Comput.*, **48**, 203–209. doi:10.2307/2007884.

[640] Kogut, J. B. (1979). An introduction to lattice gauge theory and spin systems. *Rev. Mod. Phys.*, **51**, 659–713. doi:10.1103/RevModPhys.51.659.

[641] Kokail, C., Maier, C., van Bijnen, R., et al. (2019). Self-verifying variational quantum simulation of lattice models. *Nature*, **569**, 355–360. doi:10.1038/s41586-019-1177-4. arXiv:1810.03421.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[642] Kothari, R., and O'Donnell, R. (2023). Mean estimation when you have the source code; or, quantum Monte Carlo methods. Pages 1186–1215 of: *SODA*. doi:10.1137/1.9781611977554.ch44. arXiv:2208.07544.

[643] Kovalev, A. A., and Pryadko, L. P. (2013). Fault tolerance of quantum low-density parity check codes with sublinear distance scaling. *Phys. Rev. A*, **87**. doi:10.1103/physreva.87.020304. arXiv:1208.2317.

[644] Kremenetski, V., Mejuto-Zaera, C., Cotton, S. J., and Tubman, N. M. (2021). Simulation of adiabatic quantum computing for molecular ground states. *J. Chem. Phys.*, **155**, 234106. doi:10.1063/5.0060124. arXiv:2103.12059.

[645] Kret, E. (2023). *Quantum resistance and the Signal protocol*. https://signal.org/blog/pqxdh/, accessed: 2024-06-05.

[646] Krovi, H. (2023). Improved quantum algorithms for linear and nonlinear differential equations. *Quantum*, **7**, 913. doi:10.22331/q-2023-02-02-913. arXiv:2202.01054.

[647] Kubica, A. (2018). *The ABCs of the color code: A study of topological quantum codes as toy models for fault-tolerant quantum computation and quantum phases of matter*. PhD thesis, Caltech. doi:10.7907/059V-MG69.

[648] Kubica, A., and Beverland, M. E. (2015). Universal transversal gates with color codes: A simplified approach. *Phys. Rev. A*, **91**, 032330. doi:10.1103/PhysRevA.91.032330. arXiv:1410.0069.

[649] Kubica, A., and Delfosse, N. (2023). Efficient color code decoders in $d \geq 2$ dimensions from toric code decoders. *Quantum*, **7**, 929. doi:10.22331/q-2023-02-21-929. arXiv:1905.07393.

[650] Kubica, A., and Demkowicz-Dobrzański, R. (2021). Using quantum metrological bounds in quantum error correction: A simple proof of the approximate Eastin–Knill theorem. *Phys. Rev. Lett.*, **126**, 150503. doi:10.1103/physrevlett.126.150503. arXiv:2004.11893.

[651] Kubica, A., and Preskill, J. (2019). Cellular-automaton decoders with provable thresholds for topological codes. *Phys. Rev. Lett.*, **123**, 020501. doi:10.1103/PhysRevLett.123.020501. arXiv:1809.10145.

[652] Kubica, A., and Vasmer, M. (2022). Single-shot quantum error correction with the three-dimensional subsystem toric code. *Nat. Commun.*, **13**, 6272. doi:10.1038/s41467-022-33923-4. arXiv:2106.02621.

[653] Kubica, A., Yoshida, B., and Pastawski, F. (2015). Unfolding the color code. *New J. Phys.*, **17**, 083026. doi:10.1088/1367-2630/17/8/083026. arXiv:1503.02065.

[654] Kubica, A., Haim, A., Vaknin, Y., et al. (2023). Erasure qubits: Overcoming the T_1 limit in superconducting circuits. *Phys. Rev. X*, **13**. doi:10.1103/physrevx.13.041022. arXiv:2208.05461.

[655] Kübler, J., Buchholz, S., and Schölkopf, B. (2021). The inductive bias of quantum kernels. Pages 12661–12673 of: *NeurIPS*. arXiv:2106.03747.

[656] Kueng, R., Rauhut, H., and Terstiege, U. (2017). Low rank matrix recovery from rank one measurements. *Appl. Comput. Harmon. Anal.*, **42**, 88–116. doi:10.1016/j.acha.2015.07.007. arXiv:1410.6913.

[657] Labib, F., Clader, B. D., Stamatopoulos, N., and Zeng, W. J. (2024). Quantum amplitude estimation from classical signal processing. arXiv:2405.14697.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[658] Lähde, T. A., and Meißner, U.-G. (2019). *Nuclear lattice effective field theory: An introduction*. Vol. 957. Springer.

[659] Landahl, A. J., and Ryan-Anderson, C. (2014). Quantum computing by color-code lattice surgery. arXiv:1407.5103.

[660] Lao, L., van Wee, B., Ashraf, I., et al. (2018). Mapping of lattice surgery-based quantum circuits on surface code architectures. *Quantum Sci. Technol.*, **4**, 015005. doi:10.1088/2058-9565/aadd1a. arXiv:1805.11127.

[661] Lapworth, L. (2022). A hybrid quantum-classical CFD methodology with benchmark HHL solutions. arXiv:2206.00419.

[662] Larocca, M., Czarnik, P., Sharma, K., et al. (2022). Diagnosing barren plateaus with tools from quantum optimal control. *Quantum*, **6**, 824. doi:10.22331/q-2022-09-29-824. arXiv:2105.14377.

[663] Larocca, M., Thanasilp, S., Wang, S., et al. (2024). A review of barren plateaus in variational quantum computing. arXiv:2405.00781.

[664] LaRose, R., and Coyle, B. (2020). Robust data encodings for quantum classifiers. *Phys. Rev. A*, **102**, 032420. doi:10.1103/PhysRevA.102.032420. arXiv:2003.01695.

[665] Larson, M. G., and Bengzon, F. (2013). *The finite element method: Theory, implementation, and applications*. Vol. 10. Springer Science & Business Media. doi:10.1007/978-3-642-33287-6.

[666] LeBlanc, J. P. F., Antipov, A. E., Becca, F., et al. (2015). Solutions of the two-dimensional Hubbard model: Benchmarks and results from a wide range of numerical algorithms. *Phys. Rev. X*, **5**, 041041. doi:10.1103/PhysRevX.5.041041. arXiv:1505.02290.

[667] Lee, D. (2009). Lattice simulations for few- and many-body systems. *Prog. Part. Nuc. Phys.*, **63**, 117–154. doi:10.1016/j.ppnp.2008.12.001.

[668] Lee, J., Huggins, W. J., Head-Gordon, M., and Whaley, K. B. (2018). Generalized unitary coupled cluster wave functions for quantum computation. *J. Chem. Theory Comput.*, **15**, 311–324. doi:10.1021/acs.jctc.8b01004. arXiv:1810.02327.

[669] Lee, J., Berry, D. W., Gidney, C., et al. (2021). Even more efficient quantum computations of chemistry through tensor hypercontraction. *PRX Quantum*, **2**, 030305. doi:10.1103/PRXQuantum.2.030305. arXiv:2011.03494.

[670] Lee, S., Lee, J., Zhai, H., et al. (2023). Evaluating the evidence for exponential quantum advantage in ground-state quantum chemistry. *Nat. Commun.*, **14**, 1952. doi:10.1038/s41467-023-37587-6. arXiv:2208.02199.

[671] Lee, Y. T., Sidford, A., and Wong, S. C.-W. (2015). A faster cutting plane method and its implications for combinatorial and convex optimization. Pages 1049–1065 of: *FOCS*. doi:10.1109/FOCS.2015.68. arXiv:1508.04874.

[672] Lee, Y. T., Sidford, A., and Vempala, S. S. (2018). Efficient convex optimization with membership oracles. Pages 1292–1294 of: *COLT*. arXiv:1706.07357.

[673] Lemieux, J., Duclos-Cianci, G., Sénéchal, D., and Poulin, D. (2021). Resource estimate for quantum many-body ground-state preparation on a quantum computer. *Phys. Rev. A*, **103**, 052408. doi:10.1103/PhysRevA.103.052408. arXiv:2006.04650.

[674] Leng, J., Zheng, Y., and Wu, X. (2023). A quantum-classical performance separation in nonconvex optimization. arXiv:2311.00811.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[675] Leng, J., Hickman, E., Li, J., and Wu, X. (2023). Quantum Hamiltonian descent. arXiv:2303.01471.

[676] Lenstra, A. K. (2006). Key lengths. In: *Handbook of information security*. https://infoscience.epfl.ch/handle/20.500.14299/65745, accessed: 2024-12-01.

[677] Leong, F. Y., Ewe, W.-B., and Koh, D. E. (2022). Variational quantum evolution equation solver. *Sci. Rep.*, **12**, 10817. doi:10.1038/s41598-022-14906-3. arXiv:2204.02912.

[678] Leverrier, A., and Zémor, G. (2022). Quantum Tanner codes. Pages 872–883 of: *FOCS*. doi:10.1109/FOCS54457.2022.00117. arXiv:2202.13641.

[679] Leyton, S. K., and Osborne, T. J. (2008). A quantum algorithm to solve nonlinear differential equations. arXiv:0812.4423.

[680] Li, T., Chakrabarti, S., and Wu, X. (2019). Sublinear quantum algorithms for training linear and kernel-based classifiers. Pages 3815–3824 of: *ICML*, vol. 97. arXiv:1904.02276.

[681] Li, T., Wang, C., Chakrabarti, S., and Wu, X. 2021 (5). Sublinear classical and quantum algorithms for general matrix games. Pages 8465–8473 of: *AAAI*, vol. 35. doi:10.1609/aaai.v35i10.17028. arXiv:2012.06519.

[682] Li, X., Govind, N., Isborn, C., et al. (2020). Real-time time-dependent electronic structure theory. *Chem. Rev.*, **120**, 9951–9993. doi:10.1021/acs.chemrev.0c00223.

[683] Li, X., Yin, X., Wiebe, N., et al. (2023). Potential quantum advantage for simulation of fluid dynamics. arXiv:2303.16550.

[684] Li, Y., and Benjamin, S. C. (2017). Efficient variational quantum simulator incorporating active error minimization. *Phys. Rev. X*, **7**, 021050. doi:10.1103/PhysRevX.7.021050. arXiv:1611.09301.

[685] Li, Y., and Neufeld, A. (2023). Quantum Monte Carlo algorithm for solving Black–Scholes PDEs for high-dimensional option pricing in finance and its proof of overcoming the curse of dimensionality. arXiv:2301.09241.

[686] Lim, D., and Rebentrost, P. (2024). A quantum online portfolio optimization algorithm. *Quantum Inf. Process.*, **23**, 63. doi:10.1007/s11128-024-04256-6. arXiv:2208.14749.

[687] Lin, L. (2022). Lecture notes on quantum algorithms for scientific computation. arXiv:2201.08309.

[688] Lin, L., and Tong, Y. (2020). Near-optimal ground state preparation. *Quantum*, **4**, 372. doi:10.22331/q-2020-12-14-372. arXiv:2002.12508.

[689] Lin, L., and Tong, Y. (2020). Optimal polynomial based quantum eigenstate filtering with application to solving quantum linear systems. *Quantum*, **4**, 361. doi:10.22331/q-2020-11-11-361. arXiv:1910.14596.

[690] Lin, L., and Tong, Y. (2022). Heisenberg-limited ground-state energy estimation for early fault-tolerant quantum computers. *PRX Quantum*, **3**, 010318. doi:10.1103/PRXQuantum.3.010318. arXiv:2102.11340.

[691] Linden, N., and de Wolf, R. (2021). Average-case verification of the quantum Fourier transform enables worst-case phase estimation. arXiv:2109.10215.

[692] Linden, N., Montanaro, A., and Shao, C. (2022). Quantum vs. classical algorithms for solving the heat equation. *Commun. Math. Phys.*, **395**, 601–641. doi:10.1007/s00220-022-04442-6. arXiv:2004.06516.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[693] Litinski, D. (2019). A game of surface codes: Large-scale quantum computing with lattice surgery. *Quantum*, **3**, 128. doi:10.22331/q-2019-03-05-128. arXiv:1808.02892.

[694] Litinski, D. (2019). Magic state distillation: Not as costly as you think. *Quantum*, **3**, 205. doi:10.22331/q-2019-12-02-205. arXiv:1905.06903.

[695] Litinski, D. (2023). How to compute a 256-bit elliptic curve private key with only 50 million Toffoli gates. arXiv:2306.08585.

[696] Litinski, D., and Nickerson, N. (2022). Active volume: An architecture for efficient fault-tolerant quantum computers with limited non-local connections. arXiv:2211.15465.

[697] Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Mach. Learn.*, **2**, 285–318. doi:10.1007/BF00116827.

[698] Liu, H., Ong, Y.-S., Shen, X., and Cai, J. (2020). When Gaussian process meets big data: A review of scalable GPs. *IEEE Trans. Neural Netw. Learn. Syst.*, **31**, 4405–4423. doi:10.1109/TNNLS.2019.2957109. arXiv:1807.01065.

[699] Liu, J., Li, Z., Zheng, H., et al. (2022). Towards a variational Jordan–Lee–Preskill quantum algorithm. *Mach. Learn.: Sci. Technol.*, **3**, 045030. doi:10.1088/2632-2153/aca06b. arXiv:2109.05547.

[700] Liu, J., Najafi, K., Sharma, K., et al. (2023). Analytic theory for the dynamics of wide quantum neural networks. *Phys. Rev. Lett.*, **130**, 150601. doi:10.1103/PhysRevLett.130.150601. arXiv:2203.16711.

[701] Liu, J., Liu, M., Liu, J.-P., et al. (2024). Towards provably efficient quantum algorithms for large-scale machine-learning models. *Nat. Commun.*, **15**, 434. doi:10.1038/s41467-023-43957-x. arXiv:2303.03428.

[702] Liu, J.-P., Kolden, H. Ø., Krovi, H. K., et al. (2021). Efficient quantum algorithm for dissipative nonlinear differential equations. *Proc. Natl. Acad. Sci.*, **118**, e2026805118. doi:10.1073/pnas.2026805118. arXiv:2011.03185.

[703] Liu, J.-P., An, D., Fang, D., et al. (2023). Efficient quantum algorithm for nonlinear reaction–diffusion equations and energy estimation. *Commun. Math. Phys.*, **404**, 963–1020. doi:10.1007/s00220-023-04857-9. arXiv:2205.01141.

[704] Liu, Y., Arunachalam, S., and Temme, K. (2021). A rigorous and robust quantum speed-up in supervised machine learning. *Nat. Phys.*, **17**, 1013–1017. doi:10.1038/s41567-021-01287-z. arXiv:2010.02174.

[705] Lloyd, S. (1996). Universal quantum simulators. *Science*, **273**, 1073–1078. doi:10.1126/science.273.5278.1073.

[706] Lloyd, S. (2018). Quantum approximate optimization is computationally universal. arXiv:1812.11075.

[707] Lloyd, S., Mohseni, M., and Rebentrost, P. (2013). Quantum algorithms for supervised and unsupervised machine learning. arXiv:1307.0411.

[708] Lloyd, S., Mohseni, M., and Rebentrost, P. (2014). Quantum principal component analysis. *Nat. Phys.*, **10**, 631–633. doi:10.1038/nphys3029. arXiv:1307.0401.

[709] Lloyd, S., Garnerone, S., and Zanardi, P. (2016). Quantum algorithms for topological and geometric analysis of data. *Nat. Commun.*, **7**, 1–7. doi:10.1038/ncomms10138. arXiv:1408.3106.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[710] Lloyd, S., De Palma, G., Gokler, C., et al. (2020). Quantum algorithm for nonlinear differential equations. arXiv:2011.06571.

[711] Lloyd, S., Schuld, M., Ijaz, A., et al. (2020). Quantum embeddings for machine learning. arXiv:2001.03622.

[712] Loaiza, I., and Izmaylov, A. F. (2024). Majorana tensor decomposition: A unifying framework for decompositions of fermionic Hamiltonians to linear combination of unitaries. arXiv:2407.06571.

[713] Lopes, P. C. F., Pereira, A. M. B., Clua, E. W. G., and Leiderman, R. (2022). A GPU implementation of the PCG method for large-scale image-based finite element analysis in heterogeneous periodic media. *Comput. Methods Appl. Mech. Eng.*, **399**, 115276. doi:10.1016/j.cma.2022.115276.

[714] Low, G. H. (2019). Hamiltonian simulation with nearly optimal dependence on spectral norm. Pages 491–502 of: *STOC*. doi:10.1145/3313276.3316386. arXiv:1807.03967.

[715] Low, G. H., and Chuang, I. L. (2017). Hamiltonian simulation by uniform spectral amplification. arXiv:1707.05391.

[716] Low, G. H., and Chuang, I. L. (2017). Optimal Hamiltonian simulation by quantum signal processing. *Phys. Rev. Lett.*, **118**, 010501. doi:10.1103/PhysRevLett.118.010501. arXiv:1606.02685.

[717] Low, G. H., and Chuang, I. L. (2019). Hamiltonian simulation by qubitization. *Quantum*, **3**, 163. doi:10.22331/q-2019-07-12-163. arXiv:1610.06546.

[718] Low, G. H., and Wiebe, N. (2018). Hamiltonian simulation in the interaction picture. arXiv:1805.00675.

[719] Low, G. H., Yoder, T. J., and Chuang, I. L. (2016). Methodology of resonant equiangular composite quantum gates. *Phys. Rev. X*, **6**, 041067. doi:10.1103/PhysRevX.6.041067. arXiv:1603.03996.

[720] Low, G. H., Kliuchnikov, V., and Wiebe, N. (2019). Well-conditioned multiproduct Hamiltonian simulation. arXiv:1907.11679.

[721] Low, G. H., Su, Y., Tong, Y., and Tran, M. C. (2023). Complexity of implementing Trotter steps. *PRX Quantum*, **4**, 020323. doi:10.1103/PRXQuantum.4.020323. arXiv:2211.09133.

[722] Low, G. H., Kliuchnikov, V., and Schaeffer, L. (2024). Trading T gates for dirty qubits in state preparation and unitary synthesis. *Quantum*, **8**, 1375. doi:10.22331/q-2024-06-17-1375. arXiv:1812.00954.

[723] Lowe, A., and Nayak, A. (2022). Lower bounds for learning quantum states with single-copy measurements. arXiv:2207.14438.

[724] Lu, H.-H., Klco, N., Lukens, J. M., et al. (2019). Simulations of subatomic many-body physics on a quantum frequency processor. *Phys. Rev. A*, **100**, 012320. doi:10.1103/PhysRevA.100.012320. arXiv:1810.03959.

[725] Lubasch, M., Joo, J., Moinier, P., et al. (2020). Variational quantum algorithms for nonlinear problems. *Phys. Rev. A*, **101**, 010301. doi:10.1103/PhysRevA.101.010301. arXiv:1907.09032.

[726] Lucas, A. (2014). Ising formulations of many NP problems. *Front. Phys.*, **2**. doi:10.3389/fphy.2014.00005. arXiv:1302.5843.

[727] Luo, J., and Li, L. (2024). Circuit complexity of sparse quantum state preparation. arXiv:2406.16142.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[728] Luo, Z., You, Y.-Z., Li, J., et al. (2019). Quantum simulation of the non-Fermi-liquid state of Sachdev–Ye–Kitaev model. *npj Quant. Inf.*, **5**, 53. doi:10.1038/s41534-019-0166-7. arXiv:1712.06458.

[729] Luongo, A. (2020). *Quantum algorithms for data analysis*. https://quantumalgorithms.org, accessed: 2024-11-19.

[730] Lusch, B., Kutz, J. N., and Brunton, S. L. (2018). Deep learning for universal linear embeddings of nonlinear dynamics. *Nat. Commun.*, **9**, 4950. doi:10.1038/s41467-018-07210-0. arXiv:1712.09707.

[731] Macridin, A., Spentzouris, P., Amundson, J., and Harnik, R. (2018). Digital quantum computation of fermion–boson interacting systems. *Phys. Rev. A*, **98**, 042312. doi:10.1103/PhysRevA.98.042312. arXiv:1805.09928.

[732] Macridin, A., Spentzouris, P., Amundson, J., and Harnik, R. (2018). Electron-phonon systems on a universal quantum computer. *Phys. Rev. Lett.*, **121**, 110504. doi:10.1103/PhysRevLett.121.110504. arXiv:1802.07347.

[733] Magniez, F., Nayak, A., Roland, J., and Santha, M. (2011). Search via quantum walk. *SIAM J. Comp.*, **40**, 142–164. doi:10.1137/090745854. Earlier version in *STOC'07*. arXiv:quant-ph/0608026.

[734] Magnifico, G., Felser, T., Silvi, P., and Montangero, S. (2021). Lattice quantum electrodynamics in (3+1)-dimensions at finite density with tensor networks. *Nat. Commun.*, **12**, 3600. doi:10.1038/s41467-021-23646-3. arXiv:2011.10658.

[735] Magnifico, G., Cataldi, G., Rigobello, M., et al. (2024). Tensor networks for lattice gauge theories beyond one dimension: A roadmap. arXiv:2407.03058.

[736] Mande, N. S., and de Wolf, R. (2023). Tight bounds for quantum phase estimation and related problems. Pages 81:1–81:16 of: *ESA*, vol. 274. doi:10.4230/LIPIcs.ESA.2023.81. arXiv:2305.04908.

[737] Manin, Y. I. (2007). *Mathematics as metaphor: Selected essays of Yuri I. Manin*. Vol. 20. American Mathematical Society.

[738] Manrique, D. Z., Khan, I. T., Yamamoto, K., et al. (2020). Momentum-space unitary coupled cluster and translational quantum subspace expansion for periodic systems on quantum computers. arXiv:2008.08694.

[739] Marino, R., Parisi, G., and Ricci-Tersenghi, F. (2016). The backtracking survey propagation algorithm for solving random k-SAT problems. *Nat. Commun.*, **7**, 12996. doi:10.1038/ncomms12996. arXiv:1508.05117.

[740] Maris, P., Vary, J. P., Navrátil, P., et al. (2011). Origin of the anomalous long lifetime of ^{14}C. *Phys. Rev. Lett.*, **106**, 202502. doi:10.1103/PhysRevLett.106.202502. arXiv:1101.5124.

[741] Markov, I. L., and Shi, Y. (2008). Simulating quantum computation by contracting tensor networks. *SIAM J. Comp.*, **38**, 963–981. doi:10.1137/050644756. arXiv:quant-ph/0511069.

[742] Marrero, C. O., Kieferová, M., and Wiebe, N. (2021). Entanglement-induced barren plateaus. *PRX Quantum*, **2**, 040316. doi:10.1103/PRXQuantum.2.040316. arXiv:2010.15968.

[743] Martiel, S., and Remaud, M. (2020). Practical implementation of a quantum backtracking algorithm. Pages 597–606 of: *SOFSEM*. doi:10.1007/978-3-030-38919-2_49. arXiv:1908.11291.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[744] Martyn, J. M., Rossi, Z. M., Tan, A. K., and Chuang, I. L. (2021). Grand unification of quantum algorithms. *Phys. Rev. X*, **2**, 040203. doi:10.1103/PRXQuantum.2.040203. arXiv:2105.02859.

[745] Marx, D., and Hutter, J. (2009). *Ab initio molecular dynamics: Basic theory and advanced methods*. Cambridge University Press.

[746] Maskara, N., Kubica, A., and Jochym-O'Connor, T. (2019). Advantages of versatile neural-network decoding for topological codes. *Phys. Rev. A*, **99**, 052351. doi:10.1103/PhysRevA.99.052351. arXiv:1802.08680.

[747] Mathis, S. V., Mazzola, G., and Tavernelli, I. (2020). Toward scalable simulations of lattice gauge theories on quantum computers. *Phys. Rev. D*, **102**, 094501. doi:10.1103/PhysRevD.102.094501. arXiv:2005.10271.

[748] Matsuzawa, Y., and Kurashige, Y. (2020). Jastrow-type decomposition in quantum chemistry for low-depth quantum circuits. *J. Chem. Theory Comput.*, **16**, 944–952. doi:10.1021/acs.jctc.9b00963. arXiv:1909.12410.

[749] McArdle, S. (2021). Learning from physics experiments with quantum computers: Applications in muon spectroscopy. *PRX Quantum*, **2**, 020349. doi:10.1103/PRXQuantum.2.020349. arXiv:2012.06602.

[750] McArdle, S., Mayorov, A., Shan, X., et al. (2019). Digital quantum simulation of molecular vibrations. *Chem. Sci.*, **10**, 5725–5735. doi:10.1039/C9SC01313J. arXiv:1811.04069.

[751] McArdle, S., Jones, T., Endo, S., et al. (2019). Variational ansatz-based quantum simulation of imaginary time evolution. *npj Quant. Inf.*, **5**, 75. doi:10.1038/s41534-019-0187-2. arXiv:1804.03023.

[752] McArdle, S., Endo, S., Aspuru-Guzik, A., et al. (2020). Quantum computational chemistry. *Rev. Mod. Phys.*, **92**, 015003. doi:10.1103/RevModPhys.92.015003. arXiv:1808.10402.

[753] McArdle, S., Campbell, E., and Su, Y. (2022). Exploiting fermion number in factorized decompositions of the electronic structure Hamiltonian. *Phys. Rev. A*, **105**, 012403. doi:10.1103/PhysRevA.105.012403. arXiv:2107.07238.

[754] McArdle, S., Gilyén, A., and Berta, M. (2022). Quantum state preparation without coherent arithmetic. arXiv:2210.14892.

[755] McArdle, S., Gilyén, A., and Berta, M. (2022). A streamlined quantum algorithm for topological data analysis with exponentially fewer qubits. arXiv:2209.12887.

[756] McClean, J. R., Boixo, S., Smelyanskiy, V. N., et al. (2018). Barren plateaus in quantum neural network training landscapes. *Nat. Commun.*, **9**, 1–6. doi:10.1038/s41467-018-07090-4. arXiv:1803.11173.

[757] McKeever, C., and Lubasch, M. (2023). Classically optimized Hamiltonian simulation. *Phys. Rev. Res.*, **5**, 023146. doi:10.1103/PhysRevResearch.5.023146. arXiv:2205.11427.

[758] McLachlan, A. D. (1964). A variational solution of the time-dependent Schrodinger equation. *Mol. Phys.*, **8**, 39–44. doi:10.1080/00268976400100041.

[759] Meister, R., Benjamin, S. C., and Campbell, E. T. (2022). Tailoring term truncations for electronic structure calculations using a linear combination of unitaries. *Quantum*, **6**, 637. doi:10.22331/q-2022-02-02-637. arXiv:2007.11624.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[760] Merton, R. C. (1972). An analytic derivation of the efficient portfolio frontier. *J. Financial Quant. Anal.*, **7**, 1851–1872. doi:10.2307/2329621.

[761] Meurice, Y., Sakai, R., and Unmuth-Yockey, J. (2022). Tensor lattice field theory for renormalization and quantum computing. *Rev. Mod. Phys.*, **94**, 025005. doi:10.1103/RevModPhys.94.025005. arXiv:2010.06539.

[762] Mi, X., Michailidis, A. A., Shabani, S., et al. (2024). Stable quantum-correlated many-body states through engineered dissipation. *Science*, **383**, 1332–1337. doi:10.1126/science.adh9932. arXiv:2304.13878.

[763] Miller, V. S. (1986). Use of elliptic curves in cryptography. Pages 417–426 of: *CRYPTO*. doi:10.1007/3-540-39799-X_31.

[764] Milosavljević, N., Morozov, D., and Skraba, P. (2011). Zigzag persistent homology in matrix multiplication time. Pages 216–225 of: *SoCG*. doi:10.1145/1998196.1998229.

[765] Mischaikow, K., and Nanda, V. (2013). Morse theory for filtrations and efficient computation of persistent homology. *Discrete Comput. Geom.*, **50**, 330–353. doi:10.1007/s00454-013-9529-6.

[766] Mitarai, K., Negoro, M., Kitagawa, M., and Fujii, K. (2018). Quantum circuit learning. *Phys. Rev. A*, **98**, 032309. doi:10.1103/PhysRevA.98.032309. arXiv:1803.00745.

[767] Miyamoto, K., and Kubo, K. (2021). Pricing multi-asset derivatives by finite-difference method on a quantum computer. *IEEE Trans. Quantum Eng.*, **3**, 1–25. doi:10.1109/TQE.2021.3128643. arXiv:2109.12896.

[768] Miyamoto, K., and Shiohara, K. (2020). Reduction of qubits in a quantum algorithm for Monte Carlo simulation by a pseudo-random-number generator. *Phys. Rev. A*, **102**, 022424. doi:10.1103/PhysRevA.102.022424. arXiv:1911.12469.

[769] Mizuta, K. (2023). Optimal/nearly-optimal simulation of multi-periodic time-dependent Hamiltonians. arXiv:2301.06232.

[770] Mizuta, K., and Fujii, K. (2023). Optimal Hamiltonian simulation for time-periodic systems. *Quantum*, **7**, 962. doi:10.22331/q-2023-03-28-962. arXiv:2209.05048.

[771] Mohammadisiahroudi, M., Fakhimi, R., and Terlaky, T. (2022). Efficient use of quantum linear system algorithms in interior point methods for linear optimization. arXiv:2205.01220.

[772] Mohammadisiahroudi, M., Wu, Z., Augustino, B., et al. (2022). Quantum-enhanced regression analysis using state-of-the-art QLSAs and QIPMs. Pages 375–380 of: *SEC*. doi:10.1109/SEC54971.2022.00055.

[773] Montanaro, A. (2015). Quantum speedup of Monte Carlo methods. *Proc. R. Soc. A*, **471**. doi:10.1098/rspa.2015.0301. arXiv:1504.06987.

[774] Montanaro, A. (2016). Quantum algorithms: An overview. *npj Quant. Inf.*, **2**, 15023. doi:10.1038/npjqi.2015.23. arXiv:1511.04206.

[775] Montanaro, A. (2018). Quantum-walk speedup of backtracking algorithms. *Theory Comput.*, **14**, 1–24. doi:10.4086/toc.2018.v014a015. arXiv:1509.02374.

[776] Montanaro, A. (2020). Quantum speedup of branch-and-bound algorithms. *Phys. Rev. Res.*, **2**, 013056. doi:10.1103/PhysRevResearch.2.013056. arXiv:1906.10375.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[777] Montanaro, A., and Pallister, S. (2016). Quantum algorithms and the finite element method. *Phys. Rev. A*, **93**, 032324. doi:10.1103/PhysRevA.93.032324. arXiv:1512.05903.

[778] Monteiro, R. D., and Tsuchiya, T. (2000). Polynomial convergence of primal-dual algorithms for the second-order cone program based on the MZ-family of directions. *Math. Program.*, **88**, 61–83. doi:10.1007/PL00011378.

[779] Moradi, S., Trad, D., and Innanen, K. A. (2018). Quantum computing in geophysics: Algorithms, computational costs, and future applications. In: *2018 SEG International Exposition and Annual Meeting*. doi:10.1190/segam2018-2998507.1.

[780] Morales, M. E., Biamonte, J., and Zimborás, Z. (2020). On the universality of the quantum approximate optimization algorithm. *Quantum Inf. Process.*, **19**, 1–26. doi:10.1007/s11128-020-02748-9. arXiv:1909.03123.

[781] MOSEK ApS. (2023). *MOSEK portfolio optimization cookbook: Release 1.3.0*. https://docs.mosek.com/MOSEKPortfolioCookbook-a4paper.pdf, accessed: 2023-10-04.

[782] Motlagh, D., and Wiebe, N. (2024). Generalized quantum signal processing. *PRX Quantum*, **5**, 020368. doi:10.1103/PRXQuantum.5.020368. arXiv:2308.01501.

[783] Motta, M., and Rice, J. E. (2022). Emerging quantum computing algorithms for quantum chemistry. *WIREs Comput. Mol. Sci.*, **12**, e1580. doi:10.1002/wcms.1580. arXiv:2109.02873.

[784] Motta, M., Ceperley, D. M., Chan, G. K.-L., et al. (2017). Towards the solution of the many-electron problem in real materials: Equation of state of the hydrogen chain with state-of-the-art many-body methods. *Phys. Rev. X*, **7**, 031059. doi:10.1103/PhysRevX.7.031059. arXiv:1705.01608.

[785] Motta, M., Genovese, C., Ma, F., et al. (2020). Ground-state properties of the hydrogen chain: Dimerization, insulator-to-metal transition, and magnetic phases. *Phys. Rev. X*, **10**, 031058. doi:10.1103/PhysRevX.10.031058. arXiv:1911.01618.

[786] Motta, M., Ye, E., McClean, J. R., et al. (2021). Low rank representations for quantum simulation of electronic structure. *npj Quant. Inf.*, **7**, 1–7. doi:10.1038/s41534-021-00416-z. arXiv:1808.02625.

[787] Mottonen, M., and Vartiainen, J. J. (2005). Decompositions of general quantum gates. arXiv:quant-ph/0504100.

[788] Moussa, J. E. (2016). Transversal Clifford gates on folded surface codes. *Phys. Rev. A*, **94**, 042316. doi:10.1103/PhysRevA.94.042316. arXiv:1603.02286.

[789] Moylett, A. E., Linden, N., and Montanaro, A. (2017). Quantum speedup of the traveling-salesman problem for bounded-degree graphs. *Phys. Rev. A*, **95**, 032323. doi:10.1103/PhysRevA.95.032323. arXiv:1612.06203.

[790] Mugel, S., Kuchkovsky, C., Sanchez, E., et al. (2022). Dynamic portfolio optimization with real datasets using quantum processors and quantum-inspired tensor networks. *Phys. Rev. Res.*, **4**, 013006. doi:10.1103/PhysRevResearch.4.013006. arXiv:2007.00017.

[791] Muñoz-Coreas, E., and Thapliyal, H. (2018). T-count and qubit optimized quantum circuit design of the non-restoring square root algorithm.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

ACM J. Emerg. Technol. Comput. Syst., **14**, 1–15. doi:10.1145/3264816. arXiv:1712.08254.

[792] Nagaj, D., Wocjan, P., and Zhang, Y. (2009). Fast amplification of QMA. *Quantum Inf. Comput.*, **9**, 1053–1068. doi:10.26421/QIC9.11-12. arXiv:0904.1549.

[793] Nagata, K. (2022). Finite-density lattice QCD and sign problem: Current status and open problems. *Prog. Part. Nuc. Phys.*, **127**, 103991. doi:10.1016/j.ppnp.2022.103991. arXiv:2108.12423.

[794] Nakaji, K., Bagherimehrab, M., and Aspuru-Guzik, A. (2023). qSWIFT: High-order randomized compiler for Hamiltonian simulation. arXiv:2302.14811.

[795] Nam, Y., Su, Y., and Maslov, D. (2020). Approximate quantum Fourier transform with $o(n \log(n))$ T gates. *npj Quant. Inf.*, **6**, 26. doi:10.1038/s41534-020-0257-5. arXiv:1803.04933.

[796] Navrátil, P., and Quaglioni, S. (2022). Ab initio nuclear reaction theory with applications to astrophysics. Pages 1–46 of: *Handbook of nuclear physics*. doi:10.1007/978-981-15-8818-1_7-1. arXiv:2204.01187.

[797] Nemirovski, A. S., and Todd, M. J. (2008). Interior-point methods for optimization. *Acta Numer.*, **17**, 191–234. doi:10.1017/S0962492906370018.

[798] Nesterov, Y. (2018). *Lectures on convex optimization*. Springer. doi:10.1007/978-3-319-91578-4.

[799] Nezami, S., Lin, H. W., Brown, A. R., et al. (2023). Quantum gravity in the lab. II. Teleportation by size and traversable wormholes. *PRX Quantum*, **4**, 010321. doi:10.1103/PRXQuantum.4.010321. arXiv:2102.01064.

[800] Nguyen, N., Watts, T. W., Link, B., et al. (2024). Quantum computing for corrosion-resistant materials and anti-corrosive coatings design. arXiv:2406.18759.

[801] Nielsen, M. A., and Chuang, I. L. (2000). *Quantum computation and quantum information*. Cambridge University Press. doi:10.1017/CBO9780511976667.

[802] Niroula, P., Shaydulin, R., Yalovetzky, R., et al. (2022). Constrained quantum optimization for extractive summarization on a trapped-ion quantum computer. *Sci. Rep.*, **12**, 1–14. doi:10.1038/s41598-022-20853-w. arXiv:2206.06290.

[803] Novikau, I., Startsev, E. A., and Dodin, I. Y. (2022). Quantum signal processing for simulating cold plasma waves. *Phys. Rev. A*, **105**, 062444. doi:10.1103/PhysRevA.105.062444. arXiv:2112.06086.

[804] Novo, L., and Berry, D. W. (2017). Improved Hamiltonian simulation via a truncated Taylor series and corrections. *Quantum Inf. Comput.*, **17**, 623–635. doi:10.26421/QIC17.7-8-5. arXiv:1611.10033.

[805] O'Brien, T. E., Streif, M., Rubin, N. C., et al. (2022). Efficient quantum computation of molecular forces and other energy gradients. *Phys. Rev. Res.*, **4**, 043210. doi:10.1103/PhysRevResearch.4.043210. arXiv:2111.12437.

[806] O'Brien, T. E., Ioffe, L. B., Su, Y., et al. (2022). Quantum computation of molecular structure using data from challenging-to-classically-simulate nuclear magnetic resonance experiments. *PRX Quantum*, **3**, 030345. doi:10.1103/PRXQuantum.3.030345. arXiv:2109.02163.

[807] O'Donnell, R., and Wright, J. (2016). Efficient quantum tomography. Pages 899–912 of: *STOC*. doi:10.1145/2897518.2897544. arXiv:1508.01907.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[808] O'Gorman, B., Irani, S., Whitfield, J., and Fefferman, B. (2022). Intractability of electronic structure in a fixed basis. *PRX Quantum*, **3**, 020322. doi:10.1103/PRXQuantum.3.020322. arXiv:2103.08215.

[809] O'Gorman, J., and Campbell, E. T. (2017). Quantum computation with realistic magic-state factories. *Phys. Rev. A*, **95**, 032338. doi:10.1103/PhysRevA.95.032338. arXiv:1605.07197.

[810] Oh, C., Liu, M., Alexeev, Y., et al. (2024). Classical algorithm for simulating experimental Gaussian boson sampling. *Nat. Phys.*, **20**, 1461–1468. doi:10.1038/s41567-024-02535-8. arXiv:2306.03709.

[811] Oh, C., Lim, Y., Wong, Y., et al. (2024). Quantum-inspired classical algorithms for molecular vibronic spectra. *Nat. Phys.*, **20**, 225–231. doi:10.1038/s41567-023-02308-9. arXiv:2202.01861.

[812] Oka, T., and Kitamura, S. (2019). Floquet engineering of quantum materials. *Annu. Rev. Condens. Matter Phys.*, **10**, 387–408. doi:10.1146/annurev-conmatphys-031218-013423. arXiv:1804.03212.

[813] Ollitrault, P. J., Baiardi, A., Reiher, M., and Tavernelli, I. (2020). Hardware efficient quantum algorithms for vibrational structure calculations. *Chem. Sci.*, **11**, 6842–6855. doi:10.1039/D0SC01908A. arXiv:2003.12578.

[814] Orsucci, D., and Dunjko, V. (2021). On solving classes of positive-definite quantum linear systems with quadratically improved runtime in the condition number. *Quantum*, **5**, 573. doi:10.22331/q-2021-11-08-573. arXiv:2101.11868.

[815] Orús, R. (2019). Tensor networks for complex quantum systems. *Nat. Rev. Phys.*, **1**, 538–550. doi:10.1038/s42254-019-0086-7. arXiv:1812.04011.

[816] Orús, R., and Vidal, G. (2009). Simulation of two-dimensional quantum systems on an infinite lattice revisited: Corner transfer matrix for tensor contraction. *Phys. Rev. B*, **80**, 094403. doi:10.1103/PhysRevB.80.094403. arXiv:0905.3225.

[817] Otgonbaatar, S., and Kranzlmüller, D. 2023 (7). Quantum-inspired tensor network for earth science. Pages 788–791 of: *IGARSS*. doi:10.1109/IGARSS52108.2023.10282577. arXiv:2301.07528.

[818] Otten, M., Watts, T. W., Johnson, S. D., et al. (2024). Quantum resources required for binding affinity calculations of amyloid beta. arXiv:2406.18744.

[819] Otter, N., Porter, M. A., Tillmann, U., et al. (2017). A roadmap for the computation of persistent homology. *EPJ Data Sci.*, **6**, 1–38. doi:10.1140/epjds/s13688-017-0109-5. arXiv:1506.08903.

[820] Ouyang, Y., White, D. R., and Campbell, E. T. (2020). Compilation by stochastic Hamiltonian sparsification. *Quantum*, **4**, 235. doi:10.22331/q-2020-02-27-235. arXiv:1910.06255.

[821] Oz, F., Vuppala, R. K., Kara, K., and Gaitan, F. (2022). Solving Burgers' equation with quantum computing. *Quantum Inf. Process.*, **21**, 1–13. doi:10.1007/s11128-021-03391-8.

[822] Ozols, M., Roetteler, M., and Roland, J. (2013). Quantum rejection sampling. *ACM Trans. Comput. Theory*, **5**. doi:10.1145/2493252.2493256. arXiv:1103.2774.

[823] Paini, M., and Kalev, A. (2019). An approximate description of quantum states. arXiv:1910.10543.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[824] Palmer, S., Sahin, S., Hernandez, R., et al. (2021). Quantum portfolio optimization with investment bands and target volatility. arXiv:2106.06735.

[825] Palmer, S., Karagiannis, K., Florence, A., et al. (2022). Financial index tracking via quantum computing with cardinality constraints. arXiv:2208.11380.

[826] Pan, F., and Zhang, P. (2021). Simulating the sycamore quantum supremacy circuits. arXiv:2103.03074.

[827] Panteleev, P., and Kalachev, G. (2022). Asymptotically good quantum and locally testable classical LDPC codes. Pages 375–388 of: *STOC*. doi:10.1145/3519935.3520017. arXiv:2111.03654.

[828] Park, J. W., Al-Saadon, R., MacLeod, M. K., et al. (2020). Multireference electron correlation methods: Journeys along potential energy surfaces. *Chem. Rev.*, **120**, 5878–5909. doi:10.1021/acs.chemrev.9b00496. arXiv:1911.06836.

[829] Patel, D., Tan, S. J. S., Subaşı, Y., and Sornborger, A. T. (2024). Optimal coherent quantum phase estimation via tapering. arXiv:2403.18927.

[830] Patel, R., Hsing, C.-W., Sahin, S., et al. (2022). Quantum-inspired tensor neural networks for partial differential equations. arXiv:2208.02235.

[831] Peikert, C. (2020). He gives c-sieves on the CSIDH. Pages 463–492 of: *EUROCRYPT*. doi:10.1007/978-3-030-45724-2_16. ePrint:2019/725.

[832] Penuel, J., Katabarwa, A., Johnson, P. D., et al. (2024). Feasibility of accelerating incompressible computational fluid dynamics simulations with fault-tolerant quantum computers. arXiv:2406.06323.

[833] Peruzzo, A., McClean, J., Shadbolt, P., et al. (2014). A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.*, **5**. doi:10.1038/ncomms5213. arXiv:1304.3061.

[834] Pirandola, S., Andersen, U. L., Banchi, L., et al. (2020). Advances in quantum cryptography. *Adv. Opt. Photon.*, **12**, 1012–1236. doi:10.1364/AOP.361502. arXiv:1906.01645.

[835] Plesch, M., and Brukner, i. c. v. (2011). Quantum-state preparation with universal gate decompositions. *Phys. Rev. A*, **83**, 032302. doi:10.1103/PhysRevA.83.032302. arXiv:1003.5760.

[836] Poggi, P. M. (2020). Analysis of lower bounds for quantum control times and their relation to the quantum speed limit. arXiv:2002.11147.

[837] Polkovnikov, A., Sengupta, K., Silva, A., and Vengalattore, M. (2011). Colloquium: Nonequilibrium dynamics of closed interacting quantum systems. *Rev. Mod. Phys.*, **83**, 863–883. doi:10.1103/RevModPhys.83.863. arXiv:1007.5331.

[838] Poulin, D., and Wocjan, P. (2009). Sampling from the thermal quantum Gibbs state and evaluating partition functions with a quantum computer. *Phys. Rev. Lett.*, **103**, 220502. doi:10.1103/PhysRevLett.103.220502. arXiv:0905.2199.

[839] Poulin, D., Qarry, A., Somma, R., and Verstraete, F. (2011). Quantum simulation of time-dependent Hamiltonians and the convenient illusion of Hilbert space. *Phys. Rev. Lett.*, **106**, 170501. doi:10.1103/PhysRevLett.106.170501. arXiv:1102.1360.

[840] Poulin, D., Hastings, M. B., Wecker, D., et al. (2015). The Trotter step size required for accurate quantum simulation of quantum chemistry. *Quantum Info. Comput.*, **15**, 361–384. doi:10.26421/QIC15.5-6-1. arXiv:1406.4920.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[841] Poulin, D., Kitaev, A., Steiger, D. S., et al. (2018). Quantum algorithm for spectral measurement with a lower gate count. *Phys. Rev. Lett.*, **121**, 010501. doi:10.1103/PhysRevLett.121.010501. arXiv:1711.11025.

[842] Poulsen Nautrup, H., Friis, N., and Briegel, H. J. (2017). Fault-tolerant interface between quantum memories and quantum processors. *Nat. Commun.*, **8**. doi:10.1038/s41467-017-01418-2. arXiv:1609.08062.

[843] Preskill, J. (2018). Quantum computing in the NISQ era and beyond. *Quantum*, **2**, 79. doi:10.22331/q-2018-08-06-79. arXiv:1801.00862.

[844] Preskill, J. (2019). Simulating quantum field theory with a quantum computer. doi:10.22323/1.334.0024. arXiv:1811.10085.

[845] Preskill, J. (2022). *Course information for physics 219/computer science 219 quantum computation (formerly physics 229)*. http://theory.caltech.edu/~preskill/ph229/, accessed: 2024-11-19.

[846] Proos, J., and Zalka, C. (2003). Shor's discrete logarithm quantum algorithm for elliptic curves. *Quantum Inf. Comput.*, **3**, 317–344. arXiv:0301141.

[847] Qin, M., Schafer, T., Andergassen, S., et al. (2022). The Hubbard model: A computational perspective. *Annu. Rev. Condens. Matter Phys.*, **13**, 275–302. doi:10.1146/annurev-conmatphys-090921-033948. arXiv:2104.00064.

[848] Ragavan, S. (2024). Regev factoring beyond Fibonacci: Optimizing prefactors. ePrint:2024/636.

[849] Ragavan, S., and Vaikuntanathan, V. (2024). Space-efficient and noise-robust quantum factoring. arXiv:2310.00899.

[850] Ragone, M., Bakalov, B. N., Sauvage, F., et al. (2023). A unified theory of barren plateaus for deep parametrized quantum circuits. arXiv:2309.09342.

[851] Rahmani, A., and Franz, M. (2019). Interacting Majorana fermions. *Rep. Prog. Phys.*, **82**, 084501. doi:10.1088/1361-6633/ab28ef. arXiv:1811.02593.

[852] Rajput, A., Roggero, A., and Wiebe, N. (2022). Hybridized methods for quantum simulation in the interaction picture. *Quantum*, **6**, 780. doi:10.22331/q-2022-08-17-780. arXiv:2109.03308.

[853] Rall, P. (2020). Quantum algorithms for estimating physical quantities using block encodings. *Phys. Rev. A*, **102**, 022408. doi:10.1103/PhysRevA.102.022408. arXiv:2004.06832.

[854] Rall, P. (2021). Faster coherent quantum algorithms for phase, energy, and amplitude estimation. *Quantum*, **5**, 566. doi:10.22331/q-2021-10-19-566. arXiv:2103.09717.

[855] Rall, P., and Fuller, B. (2023). Amplitude estimation from quantum signal processing. *Quantum*, **7**, 937. doi:10.22331/q-2023-03-02-937. arXiv:2207.08628.

[856] Rall, P., Wang, C., and Wocjan, P. (2023). Thermal state preparation via rounding promises. *Quantum*, **7**, 1132. doi:10.22331/q-2023-10-10-1132. arXiv:2210.01670.

[857] Ramos-Calderer, S., Pérez-Salinas, A., García-Martín, D., et al. (2021). Quantum unary approach to option pricing. *Phys. Rev. A*, **103**, 032414. doi:10.1103/PhysRevA.103.032414. arXiv:1912.01618.

[858] Rasmussen, C. E., and Williams, C. K. I. (2005). *Gaussian processes for machine learning*. The MIT Press. doi:10.7551/mitpress/3206.001.0001.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[859] Rattew, A. G., and Koczor, B. (2022). Preparing arbitrary continuous functions in quantum registers with logarithmic complexity. arXiv:2205.00519.

[860] Rattew, A. G., Sun, Y., Minssen, P., and Pistoia, M. (2021). The efficient preparation of normal distributions in quantum registers. *Quantum*, **5**, 609. doi:10.22331/q-2021-12-23-609. arXiv:2009.06601.

[861] Raussendorf, R. (2012). Key ideas in quantum error correction. *Philos. Trans. R. Soc. A*, **370**, 4541–4565. doi:10.1098/rsta.2011.0494.

[862] Raussendorf, R., and Harrington, J. (2007). Fault-tolerant quantum computation with high threshold in two dimensions. *Phys. Rev. Lett.*, **98**, 190504. doi:10.1103/PhysRevLett.98.190504. arXiv:quant-ph/0610082.

[863] Raussendorf, R., Harrington, J., and Goyal, K. (2007). Topological fault-tolerance in cluster state quantum computation. *New J. Phys.*, **9**, 199. doi:10.1088/1367-2630/9/6/199. arXiv:quant-ph/0703143.

[864] Raychowdhury, I., and Stryker, J. R. (2020). Solving Gauss's law on digital quantum computers with loop-string-hadron digitization. *Phys. Rev. Res.*, **2**, 033039. doi:10.1103/PhysRevResearch.2.033039. arXiv:1812.07554.

[865] Rebentrost, P., and Lloyd, S. (2018). Quantum computational finance: Quantum algorithm for portfolio optimization. arXiv:1811.03975.

[866] Rebentrost, P., Mohseni, M., and Lloyd, S. (2014). Quantum support vector machine for big data classification. *Phys. Rev. Lett.*, **113**, 130503. doi:10.1103/PhysRevLett.113.130503. arXiv:1307.0471.

[867] Rebentrost, P., Gupt, B., and Bromley, T. R. (2018). Quantum computational finance: Monte Carlo pricing of financial derivatives. *Phys. Rev. A*, **98**, 022321. doi:10.1103/PhysRevA.98.022321. arXiv:1805.00109.

[868] Rebentrost, P., Luongo, A., Bosch, S., and Lloyd, S. (2022). Quantum computational finance: Martingale asset pricing for incomplete markets. arXiv:2209.08867.

[869] Regev, O. (2023). An efficient quantum factoring algorithm. arXiv:2308.06572.

[870] Reiher, M., Wiebe, N., Svore, K. M., et al. (2017). Elucidating reaction mechanisms on quantum computers. *Proc. Natl. Acad. Sci.*, **114**, 7555–7560. doi:10.1073/pnas.1619152114. arXiv:1605.03590.

[871] Reiner, J. M., Zanker, S., Schwenk, I., et al. (2018). Effects of gate errors in digital quantum simulations of fermionic systems. *Quantum Sci. Technol.*, **3**. doi:10.1088/2058-9565/aad5ba. arXiv:1804.06668.

[872] Reiner, J. M., Wilhelm-Mauch, F., Schön, G., and Marthaler, M. (2019). Finding the ground state of the Hubbard model by variational methods on a quantum computer with gate errors. *Quantum Sci. Technol.*, **4**. doi:10.1088/2058-9565/ab1e85. arXiv:1811.04476.

[873] Rhodes, M., Kreshchuk, M., and Pathak, S. (2024). Exponential improvements in the simulation of lattice gauge theories using near-optimal techniques. arXiv:2405.10416.

[874] Richard, E., and Montanari, A. (2014). A statistical model for tensor PCA. In: *NeurIPS*. arXiv:1411.1076.

[875] Rivest, R. L., Shamir, A., and Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, **21**, 120–126. doi:10.1145/359340.359342.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[876] Roetteler, M., Naehrig, M., Svore, K. M., and Lauter, K. (2017). Quantum resource estimates for computing elliptic curve discrete logarithms. Pages 241–270 of: *ASIACRYPT*. doi:10.1007/978-3-319-70697-9_9. arXiv:1706.06752.

[877] Roggero, A., Li, A. C. Y., Carlson, J., et al. (2020). Quantum computing for neutrino-nucleus scattering. *Phys. Rev. D*, **101**, 074038. doi:10.1103/PhysRevD.101.074038. arXiv:1911.06368.

[878] Romero, J., and Aspuru-Guzik, A. (2021). Variational quantum generators: Generative adversarial quantum machine learning for continuous distributions. *Adv. Quantum Technol.*, **4**, 2000003. doi:10.1002/qute.202000003. arXiv:1901.00848.

[879] Romero, J., Olson, J. P., and Aspuru-Guzik, A. (2017). Quantum autoencoders for efficient compression of quantum data. *Quantum Sci. Technol.*, **2**, 045001. doi:10.1088/2058-9565/aa8072. arXiv:1612.02806.

[880] Rosenberg, E., Andersen, T. I., Samajdar, R., et al. (2024). Dynamics of magnetization at infinite temperature in a Heisenberg spin chain. *Science*, **384**, 48–53. doi:10.1126/science.adi7877. arXiv:2306.09333.

[881] Rosenberg, G., Haghnegahdar, P., Goddard, P., et al. (2016). Solving the optimal trading trajectory problem using a quantum annealer. *IEEE Journal of Selected Topics in Signal Processing*, **10**, 1053–1060. doi:10.1109/JSTSP.2016.2574703. Earlier version in *WHPCF'15*. arXiv:1508.06182.

[882] Rosenhaus, V. (2019). An introduction to the SYK model. *J. Phys. A*, **52**, 323001. doi:10.1088/1751-8121/ab2ce1. arXiv:1807.03334.

[883] Rosenthal, G. (2021). Query and depth upper bounds for quantum unitaries via Grover search. arXiv:2111.07992.

[884] Ross, N. J., and Selinger, P. (2016). Optimal ancilla-free Clifford+T approximation of z-rotations. *Quantum Inf. Comput.*, **16**, 901–953. doi:10.26421/QIC16.11-12-1. arXiv:1403.2975.

[885] Rossi, M., Asproni, L., Caputo, D., et al. (2022). Using Shor's algorithm on near term quantum computers: A reduced version. *Quantum Mach. Intell.*, **4**, 18. doi:10.1007/s42484-022-00072-2. arXiv:2112.12647.

[886] Rubin, N. C., Berry, D. W., Malone, F. D., et al. (2023). Fault-tolerant quantum simulation of materials using Bloch orbitals. *PRX Quantum*, **4**, 040303. doi:10.1103/PRXQuantum.4.040303. arXiv:2302.05531.

[887] Rubin, N. C., Berry, D. W., Kononov, A., et al. (2024). Quantum computation of stopping power for inertial fusion target design. *Proc. Natl. Acad. Sci.*, **121**, e2317772121. doi:10.1073/pnas.2317772121. arXiv:2308.12352.

[888] Saad, Y. (2003). *Iterative methods for sparse linear systems*. 2nd edn. Society for Industrial and Applied Mathematics. doi:10.1137/1.9780898718003.

[889] Sachdev, S., and Ye, J. (1993). Gapless spin-fluid ground state in a random quantum Heisenberg magnet. *Phys. Rev. Lett.*, **70**, 3339–3342. doi:10.1103/PhysRevLett.70.3339. arXiv:cond-mat/9212030.

[890] Saffman, M., Walker, T. G., and Mølmer, K. (2010). Quantum information with Rydberg atoms. *Rev. Mod. Phys.*, **82**, 2313–2363. doi:10.1103/RevModPhys.82.2313. arXiv:0909.4777.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[891] Şahinoğlu, B., and Somma, R. D. (2021). Hamiltonian simulation in the low-energy subspace. *npj Quant. Inf.*, **7**. doi:10.1038/s41534-021-00451-w. arXiv:2006.02660.

[892] Sakamoto, K., Morisaki, H., Haruna, J., et al. (2024). End-to-end complexity for simulating the Schwinger model on quantum computers. *Quantum*, **8**, 1474. doi:10.22331/q-2024-09-17-1474. arXiv:2311.17388.

[893] Salakhutdinov, R., and Larochelle, H. (2010). Efficient learning of deep Boltzmann machines. Pages 693–700 of: *AISTATS*.

[894] Sanders, Y. R., Low, G. H., Scherer, A., and Berry, D. W. (2019). Black-box quantum state preparation without arithmetic. *Phys. Rev. Lett.*, **122**, 020502. doi:10.1103/PhysRevLett.122.020502. arXiv:1807.03206.

[895] Sanders, Y. R., Berry, D. W., Costa, P. C., et al. (2020). Compilation of fault-tolerant quantum heuristics for combinatorial optimization. *PRX Quantum*, **1**, 020312. doi:10.1103/PRXQuantum.1.020312. arXiv:2007.07391.

[896] Santagati, R., Aspuru-Guzik, A., Babbush, R., et al. (2024). Drug design on quantum computers. *Nat. Phys.*, **20**, 549–557. doi:10.1038/s41567-024-02411-5. arXiv:2301.04114.

[897] Sawaya, N. P., Menke, T., Kyaw, T. H., et al. (2020). Resource-efficient digital quantum simulation of d-level systems for photonic, vibrational, and spin-s Hamiltonians. *npj Quant. Inf.*, **6**, 1–13. doi:10.1038/s41534-020-0278-0. arXiv:1909.12847.

[898] Sawaya, N. P. D., and Huh, J. (2019). Quantum algorithm for calculating molecular vibronic spectra. *J. Phys. Chem. Lett.*, **10**, 3586–3591. doi:10.1021/acs.jpclett.9b01117. arXiv:1812.10495.

[899] Sawaya, N. P. D., Paesani, F., and Tabor, D. P. (2021). Near- and long-term quantum algorithmic approaches for vibrational spectroscopy. *Phys. Rev. A*, **104**, 062419. doi:10.1103/PhysRevA.104.062419. arXiv:2009.05066.

[900] Schäfer, T., Wentzell, N., Šimkovic, F., et al. (2021). Tracking the footprints of spin fluctuations: A multimethod, multimessenger study of the two-dimensional Hubbard model. *Phys. Rev. X*, **11**, 011058. doi:10.1103/PhysRevX.11.011058. arXiv:2006.10769.

[901] Schatzki, L., Larocca, M., Nguyen, Q. T., et al. (2024). Theoretical guarantees for permutation-equivariant quantum neural networks. *npj Quant. Inf.*, **10**, 12. doi:10.1038/s41534-024-00804-1. arXiv:2210.09974.

[902] Scherer, A., Valiron, B., Mau, S.-C., et al. (2017). Concrete resource analysis of the quantum linear-system algorithm used to compute the electromagnetic scattering cross section of a 2D target. *Quantum Inf. Process.*, **16**, 1–65. doi:10.1007/s11128-016-1495-5. arXiv:1505.06552.

[903] Schmidhuber, A., and Lloyd, S. (2023). Complexity-theoretic limitations on quantum algorithms for topological data analysis. *PRX Quantum*, **4**, 040349. doi:10.1103/PRXQuantum.4.040349. arXiv:2209.14286.

[904] Schmidhuber, A., O'Donnell, R., Kothari, R., and Babbush, R. (2024). Quartic quantum speedups for planted inference. arXiv:2406.19378.

[905] Schölkopf, B., Herbrich, R., and Smola, A. J. (2001). A generalized representer theorem. Pages 416–426 of: *COLT*. doi:10.1007/3-540-44581-1_27.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[906] Scholl, P., Schuler, M., Williams, H. J., et al. (2021). Quantum simulation of 2D antiferromagnets with hundreds of Rydberg atoms. *Nature*, **595**, 233–238. doi:10.1038/s41586-021-03585-1. arXiv:2012.12268.

[907] Schollwöck, U. (2011). The density-matrix renormalization group in the age of matrix product states. *Ann. Phys.*, **326**, 96–192. doi:10.1016/j.aop.2010.09.012. arXiv:1008.3477.

[908] Schöning, U. (1999). A probabilistic algorithm for k-SAT and constraint satisfaction problems. Pages 410–414 of: *FOCS*. doi:10.1109/SFFCS.1999.814612.

[909] Schubert, A., and Mendl, C. B. (2023). Trotter error with commutator scaling for the Fermi–Hubbard model. *Phys. Rev. B*, **108**, 195105. doi:10.1103/PhysRevB.108.195105. arXiv:2306.10603.

[910] Schuch, N., and Verstraete, F. (2009). Computational complexity of interacting electrons and fundamental limitations of density functional theory. *Nat. Phys.*, **5**, 732–735. doi:10.1038/nphys1370. arXiv:0712.0483.

[911] Schuch, N., Wolf, M. M., Verstraete, F., and Cirac, J. I. (2007). Computational complexity of projected entangled pair states. *Phys. Rev. Lett.*, **98**, 140506. doi:10.1103/PhysRevLett.98.140506. arXiv:quant-ph/0611050.

[912] Schuch, N., Wolf, M. M., Verstraete, F., and Cirac, J. I. (2008). Entropy scaling and simulability by matrix product states. *Phys. Rev. Lett.*, **100**, 030504. doi:10.1103/PhysRevLett.100.030504. arXiv:0705.0292.

[913] Schuld, M. (2021). Supervised quantum machine learning models are kernel methods. arXiv:2101.11020.

[914] Schuld, M., and Killoran, N. (2019). Quantum machine learning in feature Hilbert spaces. *Phys. Rev. Lett.*, **122**, 040504. doi:10.1103/PhysRevLett.122.040504. arXiv:1803.07128.

[915] Schuld, M., and Killoran, N. (2022). Is quantum advantage the right goal for quantum machine learning? *PRX Quantum*, **3**, 030101. doi:10.1103/PRXQuantum.3.030101. arXiv:2203.01340.

[916] Schuld, M., and Petruccione, F. (2021). *Machine learning with quantum computers*. Springer. doi:10.1007/978-3-030-83098-4.

[917] Schuld, M., Sinayskiy, I., and Petruccione, F. (2016). Prediction by linear regression on a quantum computer. *Phys. Rev. A*, **94**, 022342. doi:10.1103/PhysRevA.94.022342. arXiv:1601.07823.

[918] Schuld, M., Bergholm, V., Gogolin, C., et al. (2019). Evaluating analytic gradients on quantum hardware. *Phys. Rev. A*, **99**, 032331. doi:10.1103/PhysRevA.99.032331. arXiv:1811.11184.

[919] Schuld, M., Bocharov, A., Svore, K. M., and Wiebe, N. (2020). Circuit-centric quantum classifiers. *Phys. Rev. A*, **101**, 032308. doi:10.1103/PhysRevA.101.032308. arXiv:1804.00633.

[920] Sels, D., Dashti, H., Mora, S., et al. (2020). Quantum approximate Bayesian computation for NMR model inference. *Nat. Mach. Intell.*, **2**, 396–402. doi:10.1038/s42256-020-0198-x. arXiv:1910.14221.

[921] Serbyn, M., Papić, Z., and Abanin, D. A. (2013). Universal slow growth of entanglement in interacting strongly disordered systems. *Phys. Rev. Lett.*, **110**, 260601. doi:10.1103/PhysRevLett.110.260601. arXiv:1304.4605.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[922] Setia, K., Bravyi, S., Mezzacapo, A., and Whitfield, J. D. (2019). Superfast encodings for fermionic quantum simulation. *Phys. Rev. Res.*, **1**, 033033. doi:10.1103/PhysRevResearch.1.033033. arXiv:1810.05274.

[923] Shang, H., Fan, Y., Shen, L., et al. (2023). Towards practical and massively parallel quantum computing emulation for quantum chemistry. *npj Quant. Inf.*, **9**, 33. doi:10.1038/s41534-023-00696-7. arXiv:2303.03681.

[924] Shao, C., and Montanaro, A. (2022). Faster quantum-inspired algorithms for solving linear systems. *ACM Trans. Quantum Comput.*, **3**. doi:10.1145/3520141. arXiv:2103.10309.

[925] Shao, C., and Xiang, H. (2018). Quantum circulant preconditioner for a linear system of equations. *Phys. Rev. A*, **98**, 062321. doi:10.1103/PhysRevA.98.062321. arXiv:1807.04563.

[926] Sharma, K., Cerezo, M., Cincio, L., and Coles, P. J. (2022). Trainability of dissipative perceptron-based quantum neural networks. *Phys. Rev. Lett.*, **128**, 180505. doi:10.1103/PhysRevLett.128.180505. arXiv:2005.12458.

[927] Shaw, A. F., Lougovski, P., Stryker, J. R., and Wiebe, N. (2020). Quantum algorithms for simulating the lattice Schwinger model. *Quantum*, **4**, 306. doi:10.22331/q-2020-08-10-306. arXiv:2002.11146.

[928] Shaydulin, R., Li, C., Chakrabarti, S., et al. (2023). Evidence of scaling advantage for the quantum approximate optimization algorithm on a classically intractable problem. arXiv:2308.02342.

[929] Shehab, O., Landsman, K., Nam, Y., et al. (2019). Toward convergence of effective-field-theory simulations on digital quantum computers. *Phys. Rev. A*, **100**, 062319. doi:10.1103/PhysRevA.100.062319.

[930] Shen, J., Tang, T., and Wang, L.-L. (2011). *Spectral methods: Algorithms, analysis and applications*. Vol. 41. Springer Science & Business Media. doi:10.1007/978-3-540-71041-7.

[931] Shende, V. V., Markov, I. L., and Bullock, S. S. (2004). Minimal universal two-qubit controlled-not-based circuits. *Phys. Rev. A*, **69**, 062321. doi:10.1103/PhysRevA.69.062321. arXiv:quant-ph/0308033.

[932] Shepherd, J. J., Grüneis, A., Booth, G. H., et al. (2012). Convergence of many-body wave-function expansions using a plane-wave basis: From homogeneous electron gas to solid state systems. *Phys. Rev. B*, **86**, 035111. doi:10.1103/PhysRevB.86.035111. arXiv:1202.4990.

[933] Sherrington, D., and Kirkpatrick, S. (1975). Solvable model of a spin-glass. *Phys. Rev. Lett.*, **35**, 1792. doi:10.1103/PhysRevLett.35.1792.

[934] Shokrian Zini, M., Delgado, A., dos Reis, R., et al. (2023). Quantum simulation of battery materials using ionic pseudopotentials. *Quantum*, **7**, 1049. doi:10.22331/q-2023-07-10-1049. arXiv:2302.07981.

[935] Shor, P. W. (1995). Scheme for reducing decoherence in quantum computer memory. *Phys. Rev. A*, **52**, R2493–R2496. doi:10.1103/PhysRevA.52.R2493.

[936] Shor, P. W. (1996). Fault-tolerant quantum computation. Pages 56–65 of: *FOCS*. doi:10.1109/SFCS.1996.548464. arXiv:quant-ph/9605011.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[937] Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comp.*, **26**, 1484–1509. doi:10.1137/S0097539795293172. Earlier version in *FOCS'94*. arXiv:quant-ph/9508027.

[938] Shor, P. W. (2022). The early days of quantum computation. arXiv:2208.09964.

[939] Shtanko, O., and Movassagh, R. (2021). Preparing thermal states on noiseless and noisy programmable quantum processors. arXiv:2112.14688.

[940] Simon, D. R. (1997). On the power of quantum computation. *SIAM J. Comp.*, **26**, 1474–1483. doi:10.1137/S0097539796298637. Earlier version in *FOCS'94*.

[941] Simon, S., Santagati, R., Degroote, M., et al. (2023). Improved precision scaling for simulating coupled quantum-classical dynamics. arXiv:2307.13033.

[942] Skoric, L., Browne, D. E., Barnes, K. M., et al. (2023). Parallel window decoding enables scalable fault tolerant quantum computation. *Nat. Commun.*, **14**. doi:10.1038/s41467-023-42482-1.

[943] Smolin, J. A., Smith, G., and Vargo, A. (2013). Oversimplifying quantum factoring. *Nature*, **499**, 163–165. doi:10.1038/nature12290.

[944] Somma, R., Boixo, S., and Barnum, H. (2007). Quantum simulated annealing. arXiv:0712.1008.

[945] Song, X.-Y., Jian, C.-M., and Balents, L. (2017). Strongly correlated metal built from Sachdev–Ye–Kitaev models. *Phys. Rev. Lett.*, **119**, 216601. doi:10.1103/PhysRevLett.119.216601. arXiv:1705.00117.

[946] Sparrow, C., Martín-López, E., Maraviglia, N., et al. (2018). Simulating the vibrational quantum dynamics of molecules using photonics. *Nature*, **557**, 660–667. doi:10.1038/s41586-018-0152-9.

[947] Stace, T. M., Barrett, S. D., and Doherty, A. C. (2009). Thresholds for topological codes in the presence of loss. *Phys. Rev. Lett.*, **102**, 200501. doi:10.1103/PhysRevLett.102.200501. arXiv:0904.3556.

[948] Stamatopoulos, N., and Zeng, W. J. (2023). Derivative pricing using quantum signal processing. arXiv:2307.14310.

[949] Stamatopoulos, N., Egger, D. J., Sun, Y., et al. (2020). Option pricing using quantum computers. *Quantum*, **4**, 291. doi:10.22331/q-2020-07-06-291. arXiv:1905.02666.

[950] Stamatopoulos, N., Mazzola, G., Woerner, S., and Zeng, W. J. (2022). Towards quantum advantage in financial market risk using quantum gradient algorithms. *Quantum*, **6**, 770. doi:10.22331/q-2022-07-20-770. arXiv:2111.12509.

[951] Stanisic, S., Bosse, J. L., Gambetta, F. M., et al. (2022). Observing ground-state properties of the Fermi–Hubbard model using a scalable algorithm on a quantum computer. *Nat. Commun.*, **13**, 5743. doi:10.1038/s41467-022-33335-4. arXiv:2112.02025.

[952] Steane, A. M. (1996). Error correcting codes in quantum theory. *Phys. Rev. Lett.*, **77**, 793–797. doi:10.1103/PhysRevLett.77.793.

[953] Steane, A. M. (1997). Active stabilization, quantum computation, and quantum state synthesis. *Phys. Rev. Lett.*, **78**, 2252–2255. doi:10.1103/physrevlett.78.2252. arXiv:quant-ph/9611027.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[954] Steiger, D. S., and Troyer, M. (2016). Racing in parallel: Quantum versus classical. Pages H44–010 of: *APS March Meeting Abstracts*, vol. 2016. doi:10.48660/16080019.

[955] Stetcu, I., Baroni, A., and Carlson, J. (2022). Variational approaches to constructing the many-body nuclear ground state for quantum computing. *Phys. Rev. C*, **105**, 064308. doi:10.1103/PhysRevC.105.064308. arXiv:2110.06098.

[956] Steudtner, M., Morley-Short, S., Pol, W., et al. (2023). Fault-tolerant quantum computation of molecular observables. arXiv:2303.14118.

[957] Stevenson, P. D. (2023). Comments on quantum computing in nuclear physics. *Int. J. Unconv. Comput.*, **18**, 10.

[958] Stroberg, S. R., Bogner, S. K., Hergert, H., and Holt, J. D. (2019). Nonempirical interactions for the nuclear shell model: An update. *Ann. Rev. Nucl. Part. Sci.*, **69**, 307–362. doi:10.1146/annurev-nucl-101917-021120.

[959] Strohmer, T., and Vershynin, R. (2009). A randomized Kaczmarz algorithm with exponential convergence. *J. Fourier Anal. Appl.*, **15**, 262–278. doi:10.1007/s00041-008-9030-4. arXiv:math/0702226.

[960] Stryker, J. R. (2019). Oracles for Gauss's law on digital quantum computers. *Phys. Rev. A*, **99**, 042301. doi:10.1103/PhysRevA.99.042301. arXiv:1812.01617.

[961] Su, Y. (2024). Private communication.

[962] Su, Y., Berry, D. W., Wiebe, N., et al. (2021). Fault-tolerant quantum simulations of chemistry in first quantization. *PRX Quantum*, **2**, 040332. doi:10.1103/PRXQuantum.2.040332. arXiv:2105.12767.

[963] Su, Y., Huang, H. Y., and Campbell, E. T. (2021). Nearly tight Trotterization of interacting electrons. *Quantum*, **5**, 1–58. doi:10.22331/Q-2021-07-05-495. arXiv:2012.09194.

[964] Subaşı, Y., Somma, R. D., and Orsucci, D. (2019). Quantum algorithms for systems of linear equations inspired by adiabatic quantum computing. *Phys. Rev. Lett.*, **122**, 060504. doi:10.1103/PhysRevLett.122.060504. arXiv:1805.10549.

[965] Succi, S., Itani, W., Sreenivasan, K. R., and Steijl, R. (2023). Ensemble fluid simulations on quantum computers. arXiv:2304.05410.

[966] Sugisaki, K., Toyota, K., Sato, K., et al. (2022). Adiabatic state preparation of correlated wave functions with nonlinear scheduling functions and broken-symmetry wave functions. *Commun. Chem.*, **5**, 84. doi:10.1038/s42004-022-00701-8.

[967] Sun, S.-N., Motta, M., Tazhigulov, R. N., et al. (2021). Quantum computation of finite-temperature static and dynamical properties of spin systems using quantum imaginary time evolution. *PRX Quantum*, **2**, 010317. doi:10.1103/PRXQuantum.2.010317. arXiv:2009.03542.

[968] Sun, X., Tian, G., Yang, S., et al. (2023). Asymptotically optimal circuit depth for quantum state preparation and general unitary synthesis. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 1–1. doi:10.1109/TCAD.2023.3244885. arXiv:2108.06150.

[969] Sünderhauf, C., Campbell, E., and Camps, J. (2023). Block-encoding structured matrices for data input in quantum computing. arXiv:2302.10949.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[970] Susskind, L. (1977). Lattice fermions. *Phys. Rev. D*, **16**, 3031–3039. doi:10.1103/PhysRevD.16.3031.

[971] Suykens, J., De Brabanter, J., Lukas, L., and Vandewalle, J. (2002). Weighted least squares support vector machines: Robustness and sparse approximation. *Neurocomputing*, **48**, 85–105. doi:10.1016/S0925-2312(01)00644-0.

[972] Suykens, J. A. K., and Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Process. Lett.*, **9**, 293–300. doi:10.1023/A:1018628609742.

[973] Sweke, R., Kesselring, M. S., van Nieuwenburg, E. P. L., and Eisert, J. (2020). Reinforcement learning decoders for fault-tolerant quantum computation. *Mach. Learn.: Sci. Technol.*, **2**, 025005. doi:10.1088/2632-2153/abc609. arXiv:1810.07207.

[974] Szegedy, M. (2004). Quantum speed-up of Markov chain based algorithms. Pages 32–41 of: *FOCS*. doi:10.1109/FOCS.2004.53. arXiv:quant-ph/0401053.

[975] Ta-Shma, A. (2013). Inverting well conditioned matrices in quantum logspace. Pages 881–890 of: *STOC*. doi:10.1145/2488608.2488720.

[976] Tang, E. (2019). A quantum-inspired classical algorithm for recommendation systems. Pages 217–228 of: *STOC*. doi:10.1145/3313276.3316310. arXiv:1807.04271.

[977] Tang, E. (2021). Quantum principal component analysis only achieves an exponential speedup because of its state preparation assumptions. *Phys. Rev. Lett.*, **127**, 060503. doi:10.1103/PhysRevLett.127.060503. arXiv:1811.00414.

[978] Tang, E. (2022). Dequantizing algorithms to understand quantum advantage in machine learning. *Nat. Rev. Phys.*, **4**, 692–693. doi:10.1038/s42254-022-00511-w.

[979] Tang, E. (2023). *Quantum machine learning without any quantum*. PhD thesis, University of Washington.

[980] Tang, E., and Tian, K. (2023). A CS guide to the quantum singular value transformation. arXiv:2302.14324.

[981] Tang, H., Pal, A., Wang, T.-Y., et al. (2021). Quantum computation for pricing the collateralized debt obligations. *Quantum Eng.*, **3**, e84. doi:10.1002/que2.84. arXiv:2008.04110.

[982] Taube, A. G., and Bartlett, R. J. (2006). New perspectives on unitary coupled-cluster theory. *Int. J. Quantum Chem.*, **106**, 3393–3401. doi:10.1002/qua.21198.

[983] Tazhigulov, R. N., Sun, S.-N., Haghshenas, R., et al. (2022). Simulating models of challenging correlated molecules and materials on the sycamore quantum processor. *PRX Quantum*, **3**, 040318. doi:10.1103/PRXQuantum.3.040318. arXiv:2203.15291.

[984] Temme, K., Osborne, T. J., Vollbrecht, K. G., et al. (2011). Quantum Metropolis sampling. *Nature*, **471**, 87–90. doi:10.1038/nature09770. arXiv:0911.3635.

[985] Terhal, B. M. (2015). Quantum error correction for quantum memories. *Rev. Mod. Phys.*, **87**, 307–346. doi:10.1103/revmodphys.87.307. arXiv:1302.3428.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[986] Terhal, B. M., and Burkard, G. (2005). Fault-tolerant quantum computation for local non-Markovian noise. *Phys. Rev. A*, **71**, 012336. doi:10.1103/physreva.71.012336. arXiv:quant-ph/0402104.

[987] Tews, I., Davoudi, Z., Ekström, A., et al. (2020). New ideas in constraining nuclear forces. *J. Phys. G*, **47**, 103001. doi:10.1088/1361-6471/ab9079. arXiv:2001.03334.

[988] Thanasilp, S., Wang, S., Cerezo, M., and Holmes, Z. (2022). Exponential concentration and untrainability in quantum kernel methods. arXiv:2208.11060.

[989] Thomas, P. S., Carrington Jr., T., Agarwal, J., and Schaefer III, H. F. (2018). Using an iterative eigensolver and intertwined rank reduction to compute vibrational spectra of molecules with more than a dozen atoms: Uracil and naphthalene. *J. Chem. Phys.*, **149**, 064108. doi:10.1063/1.5039147.

[990] Tong, Y., An, D., Wiebe, N., and Lin, L. (2021). Fast inversion, preconditioned quantum linear system solvers, fast Green's-function computation, and fast evaluation of matrix functions. *Phys. Rev. A*, **104**, 032422. doi:10.1103/PhysRevA.104.032422. arXiv:2008.13295.

[991] Tong, Y., Albert, V. V., McClean, J. R., et al. (2022). Provably accurate simulation of gauge theories and bosonic systems. *Quantum*, **6**, 816. doi:10.22331/q-2022-09-22-816. arXiv:2110.06942.

[992] Torlai, G., and Melko, R. G. (2017). Neural decoder for topological codes. *Phys. Rev. Lett.*, **119**, 030501. doi:10.1103/PhysRevLett.119.030501. arXiv:1610.04238.

[993] Tran, M. C., Guo, A. Y., Su, Y., et al. (2019). Locality and digital quantum simulation of power-law interactions. *Phys. Rev. X*, **9**, 031006. doi:10.1103/PhysRevX.9.031006. arXiv:1808.05225.

[994] Tran, M. C., Su, Y., Carney, D., and Taylor, J. M. (2021). Faster digital quantum simulation by symmetry protection. *PRX Quantum*, **2**, 010323. doi:10.1103/PRXQuantum.2.010323. arXiv:2006.16248.

[995] Trenev, D., Ollitrault, P. J., Harwood, S. M., et al. (2023). Refining resource estimation for the quantum computation of vibrational molecular spectra through Trotter error analysis. arXiv:2311.03719.

[996] Trottenberg, U., Oosterlee, C. W., and Schuller, A. (2000). *Multigrid*. Elsevier.

[997] Troyer, M., and Wiese, U.-J. (2005). Computational complexity and fundamental limitations to fermionic quantum Monte Carlo simulations. *Phys. Rev. Lett.*, **94**, 170201. doi:10.1103/PhysRevLett.94.170201. arXiv:cond-mat/0408370.

[998] Tubman, N. M., Mejuto-Zaera, C., Epstein, J. M., et al. (2018). Postponing the orthogonality catastrophe: Efficient state preparation for electronic structure simulations on quantum devices. arXiv:1809.05523.

[999] Tuckett, D. K., Bartlett, S. D., and Flammia, S. T. (2018). Ultrahigh error threshold for surface codes with biased noise. *Phys. Rev. Lett.*, **120**, 050505. doi:10.1103/PhysRevLett.120.050505. arXiv:1708.08474.

[1000] Turro, F., Chistolini, T., Hashim, A., et al. (2023). Demonstration of a quantum-classical coprocessing protocol for simulating nuclear reactions. *Phys. Rev. A*, **108**, 032417. doi:10.1103/PhysRevA.108.032417.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[1001] Valahu, C. H., Olaya-Agudelo, V. C., MacDonell, R. J., et al. (2023). Direct observation of geometric-phase interference in dynamics around a conical intersection. *Nat. Chem.*, **15**, 1503–1508. doi:10.1038/s41557-023-01300-3. arXiv:2211.07320.

[1002] van Vreumingen, D., Neukart, F., Von Dollen, D., et al. (2019). Quantum-assisted finite-element design optimization. arXiv:1908.03947.

[1003] Varbanov, B. M., Serra-Peralta, M., Byfield, D., and Terhal, B. M. (2023). Neural network decoder for near-term surface-code experiments. arXiv:2307.03280.

[1004] Vasmer, M., and Kubica, A. (2022). Morphing quantum codes. *PRX Quantum*, **3**, 030319. doi:10.1103/PRXQuantum.3.030319. arXiv:2112.01446.

[1005] Vasmer, M., Browne, D. E., and Kubica, A. (2021). Cellular automaton decoders for topological quantum codes with noisy measurements and beyond. *Sci. Rep.*, **11**, 2027. doi:10.1038/s41598-021-81138-2. arXiv:2004.07247.

[1006] Veis, L., and Pittner, J. (2014). Adiabatic state preparation study of methylene. *J. Chem. Phys.*, **140**, 214111. doi:10.1063/1.4880755. arXiv:1401.3186.

[1007] Venkateswaran, R., and O'Donnell, R. (2020). Quantum approximate counting with nonadaptive Grover iterations. arXiv:2010.04370.

[1008] Verdon, G., Broughton, M., and Biamonte, J. (2017). A quantum algorithm to train neural networks using low-depth circuits. arXiv:1712.05304.

[1009] Verdon, G., Pye, J., and Broughton, M. (2018). A universal training algorithm for quantum deep learning. arXiv:1806.09729.

[1010] Verstraete, F., and Cirac, J. I. (2005). Mapping local Hamiltonians of fermions to local Hamiltonians of spins. *J. Stat. Mech. Theory Exp.*, **2005**, P09012. doi:10.1088/1742-5468/2005/09/P09012. arXiv:cond-mat/0508353.

[1011] Wan, K. (2021). Exponentially faster implementations of Select(H) for fermionic Hamiltonians. *Quantum*, **5**. doi:10.22331/q-2021-01-12-380. arXiv:2004.04170.

[1012] Wan, K., and Kim, I. (2020). Fast digital methods for adiabatic state preparation. arXiv:2004.04164.

[1013] Wan, K., Berta, M., and Campbell, E. T. (2022). Randomized quantum algorithm for statistical phase estimation. *Phys. Rev. Lett.*, **129**, 030503. doi:10.1103/PhysRevLett.129.030503. arXiv:2110.12071.

[1014] Wan, K. H., Dahlsten, O., Kristjánsson, H., et al. (2017). Quantum generalisation of feedforward neural networks. *npj Quant. Inf.*, **3**, 36. doi:10.1038/s41534-017-0032-4. arXiv:1612.01045.

[1015] Wang, C. S., Curtis, J. C., Lester, B. J., et al. (2020). Efficient multiphoton sampling of molecular vibronic spectra on a superconducting bosonic processor. *Phys. Rev. X*, **10**, 021060. doi:10.1103/PhysRevX.10.021060. arXiv:1908.03598.

[1016] Wang, C. S., Frattini, N. E., Chapman, B. J., et al. (2023). Observation of wavepacket branching through an engineered conical intersection. *Phys. Rev. X*, **13**, 011008. doi:10.1103/PhysRevX.13.011008. arXiv:2202.02364.

[1017] Wang, D. S., Fowler, A. G., and Hollenberg, L. C. L. (2011). Surface code quantum computing with error rates over 1%. *Phys. Rev. A*, **83**, 020302. doi:10.1103/PhysRevA.83.020302.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[1018] Wang, J., Dong, Y., and Lin, L. (2022). On the energy landscape of symmetric quantum signal processing. *Quantum*, **6**, 850. doi:10.22331/q-2022-11-03-850. arXiv:2110.04993.

[1019] Wang, R., Kashinath, K., Mustafa, M., et al. (2020). Towards physics-informed deep learning for turbulent flow prediction. Pages 1457–1466 of: *KDD*. doi:10.1145/3394486.3403198. arXiv:1911.08655.

[1020] Wang, S., Wang, Z., Cui, G., et al. (2021). Fast black-box quantum state preparation based on linear combination of unitaries. *Quantum Inf. Process.*, **20**, 1–14. doi:10.1007/s11128-021-03203-z. arXiv:2105.06230.

[1021] Wang, S., Fontana, E., Cerezo, M., et al. (2021). Noise-induced barren plateaus in variational quantum algorithms. *Nat. Commun.*, **12**, 1–11. doi:10.1038/s41467-021-27045-6. arXiv:2007.14384.

[1022] Wang, S., McArdle, S., and Berta, M. (2024). Qubit-efficient randomized quantum algorithms for linear algebra. *PRX Quantum*, **5**, 020324. doi:10.1103/PRXQuantum.5.020324. arXiv:2302.01873.

[1023] Wang, Z., Wei, S., Long, G.-L., and Hanzo, L. (2022). Variational quantum attacks threaten advanced encryption standard based symmetric cryptography. *Sci. China Inf. Sci.*, **65**, 200503. doi:10.1007/s11432-022-3511-5. arXiv:2205.03529.

[1024] Wanner, G., and Hairer, E. (1993). *Solving ordinary differential equations I*. Vol. 375. Springer Berlin, Heidelberg. doi:10.1007/978-3-540-78862-1.

[1025] Wanner, G., and Hairer, E. (1996). *Solving ordinary differential equations II*. Vol. 375. Springer Berlin, Heidelberg. doi:10.1007/978-3-642-05221-7.

[1026] Washington, L. C. (2008). *Elliptic curves: Number theory and cryptography, second edition*. Chapman and Hall/CRC.

[1027] Watkins, J., Wiebe, N., Roggero, A., and Lee, D. (2022). Time-dependent Hamiltonian simulation using discrete clock constructions. arXiv:2203.11353.

[1028] Watson, J. D., Bringewatt, J., Shaw, A. F., et al. (2023). Quantum algorithms for simulating nuclear effective field theories. arXiv:2312.05344.

[1029] Watts, T. W., Otten, M., Necaise, J. T., et al. (2024). Fullerene-encapsulated cyclic ozone for the next generation of nano-sized propellants via quantum computation. arXiv:2408.13244.

[1030] Webber, M., Elfving, V., Weidt, S., and Hensinger, W. K. (2022). The impact of hardware specifications on reaching quantum advantage in the fault tolerant regime. *AVS Quantum Sci.*, **4**, 013801. doi:10.1116/5.0073075. arXiv:2108.12371.

[1031] Wecker, D., Hastings, M. B., Wiebe, N., et al. (2015). Solving strongly correlated electron models on a quantum computer. *Phys. Rev. A*, **92**, 062318. doi:10.1103/PhysRevA.92.062318. arXiv:1506.05135.

[1032] Wecker, D., Hastings, M. B., and Troyer, M. (2016). Training a quantum optimizer. *Phys. Rev. A*, **94**, 022309. doi:10.1103/PhysRevA.94.022309. arXiv:1605.05370.

[1033] Wein, A. S., El Alaoui, A., and Moore, C. (2019). The Kikuchi hierarchy and tensor PCA. Pages 1446–1468 of: *FOCS*. doi:10.1109/FOCS.2019.000-2. arXiv:1904.03858.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[1034] Whitfield, J. D., Biamonte, J., and Aspuru-Guzik, A. (2011). Simulation of electronic structure Hamiltonians using quantum computers. *Mol. Phys.*, **109**, 735–750. doi:10.1080/00268976.2011.552441. arXiv:1001.3855.

[1035] Whitfield, J. D., Love, P. J., and Aspuru-Guzik, A. (2013). Computational complexity in electronic structure. *Phys. Chem. Chem. Phys.*, **15**, 397–411. doi:10.1039/C2CP42695A. arXiv:1208.3334.

[1036] Whitlow, J., Jia, Z., Wang, Y., et al. (2023). Quantum simulation of conical intersections using trapped ions. *Nat. Chem.*, **15**, 1509–1514. doi:10.1038/s41557-023-01303-0. arXiv:2211.07319.

[1037] Wiebe, N., and Wossnig, L. (2019). Generative training of quantum Boltzmann machines with hidden units. arXiv:1905.09902.

[1038] Wiebe, N., Berry, D., Høyer, P., and Sanders, B. C. (2010). Higher order decompositions of ordered operator exponentials. *J. Phys. A*, **43**, 065203. doi:10.1088/1751-8113/43/6/065203. arXiv:0812.0562.

[1039] Wiebe, N., Braun, D., and Lloyd, S. (2012). Quantum algorithm for data fitting. *Phys. Rev. Lett.*, **109**, 050505. doi:10.1103/PhysRevLett.109.050505. arXiv:1204.5242.

[1040] Wiebe, N., Kapoor, A., and Svore, K. M. (2015). Quantum nearest-neighbor algorithms for machine learning. *Quantum Inf. Comput.*, **15**, 318–358. doi:10.26421/QIC15.3-4. arXiv:1401.2142.

[1041] Wiebe, N., Kapoor, A., and Svore, K. M. (2016). Quantum deep learning. *Quantum Inf. Comput.*, **16**, 541–587. doi:10.26421/QIC16.7-8-1. arXiv:1412.3489.

[1042] Wiebe, N., Bocharov, A., Smolensky, P., et al. (2019). Quantum language processing. arXiv:1902.05162.

[1043] Wierichs, D., Izaac, J., Wang, C., and Lin, C. Y.-Y. (2022). General parameter-shift rules for quantum gradients. *Quantum*. doi:10.22331/q-2022-03-30-677. arXiv:2107.12390.

[1044] Wilde, M. M. (2017). *Quantum information theory*. 2nd edn. Cambridge University Press. doi:10.1017/9781316809976. arXiv:1106.1445.

[1045] Wilkinson, J. M., and Blundell, S. J. (2020). Information and decoherence in a muon-fluorine coupled system. *Phys. Rev. Lett.*, **125**, 087201. doi:10.1103/PhysRevLett.125.087201. arXiv:2003.02762.

[1046] Williams, K. T., Yao, Y., Li, J., et al. (2020). Direct comparison of many-body methods for realistic electronic Hamiltonians. *Phys. Rev. X*, **10**, 011041. doi:10.1103/PhysRevX.10.011041. arXiv:1910.00045.

[1047] Williams, R. (2005). A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, **348**, 357–365. doi:10.1016/j.tcs.2005.09.023. Earlier version in *ICALP'04*.

[1048] Wocjan, P., and Temme, K. (2023). Szegedy walk unitaries for quantum maps. *Commun. Math. Phys.*, **402**, 3201–3231. doi:10.1007/s00220-023-04797-4. arXiv:2107.07365.

[1049] Woerner, S., and Egger, D. J. (2019). Quantum risk analysis. *npj Quant. Inf.*, **5**, 15. doi:10.1038/s41534-019-0130-6. arXiv:1806.06893.

[1050] Wolf, M. M. (2012). *Quantum channels & operations: Guided tour*. https://mediatum.ub.tum.de/download/1701036/1701036.pdf, accessed: 2023-09-30.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[1051] de Wolf, R. (2019). Quantum computing: Lecture notes. arXiv:1907.09415.
[1052] Wright, M. (2005). The interior-point revolution in optimization: History, recent developments, and lasting consequences. *Bull. AMS*, **42**, 39–56. doi:10.1090/S0273-0979-04-01040-7.
[1053] Wright, S. J. (1997). *Primal-dual interior-point methods*. Society for Industrial and Applied Mathematics. doi:10.1137/1.9781611971453.
[1054] Wu, D., Rossi, R., Vicentini, F., et al. (2023). Variational benchmarks for quantum many-body problems. arXiv:2302.04919.
[1055] Wu, H.-C., Wang, J., and Li, X. (2024). Quantum algorithms for nonlinear dynamics: Revisiting Carleman linearization with no dissipative conditions. arXiv:2405.12714.
[1056] Wu, L.-A., Byrd, M. S., and Lidar, D. A. (2002). Polynomial-time simulation of pairing models on a quantum computer. *Phys. Rev. Lett.*, **89**, 057904. doi:10.1103/PhysRevLett.89.057904. arXiv:quant-ph/0108110.
[1057] Wu, Y., Kolkowitz, S., Puri, S., and Thompson, J. D. (2022). Erasure conversion for fault-tolerant quantum computing in alkaline earth Rydberg atom arrays. *Nat. Commun.*, **13**, 4657. doi:10.1038/s41467-022-32094-6. arXiv:2201.03540.
[1058] Xu, F., Ma, X., Zhang, Q., et al. (2020). Secure quantum key distribution with realistic devices. *Rev. Mod. Phys.*, **92**, 025002. doi:10.1103/RevModPhys.92.025002. arXiv:1903.09051.
[1059] Xu, S., Susskind, L., Su, Y., and Swingle, B. (2020). A sparse model of quantum holography. arXiv:2008.02303.
[1060] Xu, X., Sun, J., Endo, S., et al. (2021). Variational algorithms for linear algebra. *Sci. Bull.*, **66**, 2181–2188. doi:10.1016/j.scib.2021.06.023. arXiv:1909.03898.
[1061] Xue, C., Xu, X.-F., Wu, Y.-C., and Guo, G.-P. (2022). Quantum algorithm for solving a quadratic nonlinear system of equations. *Phys. Rev. A*, **106**, 032427. doi:10.1103/PhysRevA.106.032427.
[1062] Yalovetzky, R., Minssen, P., Herman, D., and Pistoia, M. (2021). NISQ-HHL: Portfolio optimization for near-term quantum hardware. arXiv:2110.15958.
[1063] Yamada, S., Imamura, T., and Machida, M. (2005). 16.447 TFlops and 159-billion-dimensional exact-diagonalization for trapped Fermion–Hubbard model on the Earth simulator. Pages 44–44 of: *SC*. doi:10.1109/SC.2005.1.
[1064] Yamasaki, H., Subramanian, S., Sonoda, S., and Koashi, M. (2020). Learning with optimized random features: Exponential speedup by quantum machine learning without sparsity and low-rank assumptions. Pages 13674–13687 of: *NeurIPS*, vol. 33. arXiv:2004.10756.
[1065] Ying, L. (2022). Stable factorization for phase factors of quantum signal processing. *Quantum*, **6**, 842. doi:10.22331/q-2022-10-20-842. arXiv:2202.02671.
[1066] Yoder, T. J., and Kim, I. H. (2017). The surface code with a twist. *Quantum*, **1**, 2. doi:10.22331/q-2017-04-25-2. arXiv:1612.04795.
[1067] Yoder, T. J., Low, G. H., and Chuang, I. L. (2014). Fixed-point quantum search with an optimal number of queries. *Phys. Rev. Lett.*, **113**, 210501. doi:10.1103/PhysRevLett.113.210501. arXiv:1409.3305.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[1068] Yoshioka, N., Sato, T., Nakagawa, Y. O., et al. (2022). Variational quantum simulation for periodic materials. *Phys. Rev. Res.*, **4**, 013052. doi:10.1103/PhysRevResearch.4.013052. arXiv:2008.09492.

[1069] Yoshioka, N., Okubo, T., Suzuki, Y., et al. (2024). Hunting for quantum-classical crossover in condensed matter problems. *npj Quant. Inf.*, **10**, 45. doi:10.1038/s41534-024-00839-4. arXiv:2210.14109.

[1070] You, X., Chakrabarti, S., Chen, B., and Wu, X. (2023). Analyzing convergence in quantum neural networks: Deviations from neural tangent kernels. Pages 40199–40224 of: *ICML*, vol. 202. arXiv:2303.14844.

[1071] Young, A. P., Knysh, S., and Smelyanskiy, V. N. (2010). First-order phase transition in the quantum adiabatic algorithm. *Phys. Rev. Lett.*, **104**, 020502. doi:10.1103/PhysRevLett.104.020502. arXiv:0910.1378.

[1072] Yu, H., Lu, D., Wu, Q., and Wei, T.-C. (2022). Geometric quantum adiabatic methods for quantum chemistry. *Phys. Rev. Res.*, **4**, 033045. doi:10.1103/PhysRevResearch.4.033045. arXiv:2112.15186.

[1073] Yuan, P., and Zhang, S. (2023). Optimal (controlled) quantum state preparation and improved unitary synthesis by quantum circuits with any number of ancillary qubits. *Quantum*, **7**, 956. doi:10.22331/q-2023-03-20-956. arXiv:2202.11302.

[1074] Yuan, X., Endo, S., Zhao, Q., et al. (2019). Theory of variational quantum simulation. *Quantum*, **3**, 191. doi:10.22331/q-2019-10-07-191. arXiv:1812.08767.

[1075] Yuen, H. (2023). An improved sample complexity lower bound for (fidelity) quantum state tomography. *Quantum*, **7**, 890. doi:10.22331/q-2023-01-03-890. arXiv:2206.11185.

[1076] Yung, M.-H., and Aspuru-Guzik, A. (2012). A quantum-quantum Metropolis algorithm. *Proc. Natl. Acad. Sci.*, **109**, 754–759. doi:10.1073/pnas.1111758109. arXiv:1011.1468.

[1077] Zaletel, M. P., and Pollmann, F. (2020). Isometric tensor network states in two dimensions. *Phys. Rev. Lett.*, **124**, 037201. doi:10.1103/PhysRevLett.124.037201. arXiv:1902.05100.

[1078] Zeng, B., Cross, A., and Chuang, I. L. (2011). Transversality versus universality for additive quantum codes. *IEEE Trans. Inf. Theory*, **57**, 6272–6284. doi:10.1109/tit.2011.2161917. arXiv:0706.1382.

[1079] Zhang, C., and Li, T. (2021). Escape saddle points by a simple gradient-descent based algorithm. Pages 8545–8556 of: *NeurIPS*. arXiv:2111.14069.

[1080] Zhang, C., and Li, T. (2023). Quantum lower bounds for finding stationary points of nonconvex functions. Pages 41268–41299 of: *ICML*, vol. 202. arXiv:2212.03906.

[1081] Zhang, C., Leng, J., and Li, T. (2021). Quantum algorithms for escaping from saddle points. *Quantum*, **5**, 529. doi:10.22331/q-2021-08-20-529. arXiv:2007.10253.

[1082] Zhang, J., Feng, F., and Zhang, Q. (2021). Quantum method for finite element simulation of electromagnetic problems. Pages 120–123 of: *IMS*. doi:10.1109/IMS19712.2021.9574852.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

[1083] Zhang, K., Yu, K., and Korepin, V. (2022). Quantum search on noisy intermediate-scale quantum devices. *Europhys. Lett.*, **140**, 18002. doi:10.1209/0295-5075/ac90e6. arXiv:2202.00122.

[1084] Zhang, X.-M., and Yuan, X. (2024). Circuit complexity of quantum access models for encoding classical data. *npj Quant. Inf.*, **10**, 42. doi:10.1038/s41534-024-00835-8. arXiv:2311.11365.

[1085] Zhang, X.-M., Li, T., and Yuan, X. (2022). Quantum state preparation with optimal circuit depth: Implementations and applications. *Phys. Rev. Lett.*, **129**, 230504. doi:10.1103/PhysRevLett.129.230504. arXiv:2201.11495.

[1086] Zhao, Q., and Yuan, X. (2021). Exploiting anticommutation in Hamiltonian simulation. *Quantum*, **5**, 534. doi:10.22331/q-2021-08-31-534. arXiv:2103.07988.

[1087] Zhao, Q., Zhou, Y., Shaw, A. F., et al. (2022). Hamiltonian simulation with random inputs. *Phys. Rev. Lett.*, **129**, 270502. doi:10.1103/PhysRevLett.129.270502. arXiv:2111.04773.

[1088] Zhao, Z., Fitzsimons, J. K., Osborne, M. A., et al. (2019). Quantum algorithms for training Gaussian processes. *Phys. Rev. A*, **100**, 012304. doi:10.1103/PhysRevA.100.012304. arXiv:1803.10520.

[1089] Zhao, Z., Fitzsimons, J. K., and Fitzsimons, J. F. (2019). Quantum-assisted Gaussian process regression. *Phys. Rev. A*, **99**, 052331. doi:10.1103/PhysRevA.99.052331. arXiv:1512.03929.

[1090] Zhou, H., Zhao, C., Cain, M., et al. (2024). Algorithmic fault tolerance for fast quantum computing. arXiv:2406.17653.

[1091] Zhou, L., Basso, J., and Mei, S. (2024). Statistical estimation in the spiked tensor model via the quantum approximate optimization algorithm. arXiv:2402.19456.

[1092] Zobel, J. P., and González, L. (2021). The quest to simulate excited-state dynamics of transition metal complexes. *JACS Au*, **1**, 1116–1140. doi:10.1021/jacsau.1c00252. arXiv:2106.11912.

[1093] Zoufal, C., Lucchi, A., and Woerner, S. (2021). Variational quantum Boltzmann machines. *Quantum Mach. Intell.*, **3**, 7. doi:10.1007/s42484-020-00033-7. arXiv:2006.06004.

[1094] Žutić, I., Fabian, J., and Das Sarma, S. (2004). Spintronics: Fundamentals and applications. *Rev. Mod. Phys.*, **76**, 323–410. doi:10.1103/RevModPhys.76.323. arXiv:cond-mat/0405528.

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Index

#P-hard, 45
adiabatic state preparation, *see* quantum adiabatic algorithm
AES (Advanced Encryption Standard), 108
amplitude amplification, 205, 235–238, *see also* Grover search
  applications of, 64, 104, 108, 165, 175, 265, 288, 302
  fixed-point, 209, 237, 245, 254
  oblivious, 204, 219, 222, 237
  on top of phase estimation, 9, 36, 59, 73, 171, 229
  on top of QAOA, 71
  variable-time, 273
  with Gibbs sampling, 80, 244
amplitude estimation, 205, 233, 238–242, 279, *see also* overlap estimation
  applications of, 19, 59, 80, 118, 131, 143, 150, 158, 165, 175, 307
  nondestructive, 241
  unbiased, 241
analog quantum simulator, 14, 23, 44, 49, 56
approximate counting, *see* amplitude estimation

backlog problem, 318, 321
backtracking algorithms, 68
barren plateau, 96, 184, 283
Bernstein–Vazirani algorithm, ix, 277
Betti number, 173–179
block-encoding, 189–198
  for Hamiltonian access, 7, 18, 34, 59, 211, 244
  of an LCU, 7, 19, 26, 34, 156, 192, 197, 223, 264
  of classical data, 84, 135, 152, 155, 158, 191, 193, 248, 266–269, 273, 295
  of Gram matrices, 156, 191, 267
  of observables, 10, 38
  of sparse matrices, 85, 153, 160, 172, 191, 269, 273, 297
Boltzmann machine (BM), 161
  quantum (QBM), 164, 249
  restricted (RBM), 162
Born–Oppenheimer approximation, 30, 45
BQP, 342
BQP-complete, 20, 43, 56, 168, 210, 343
BQP-hard, 125, 172, 183, 307, 343
branch-and-bound method, 68, 75, 134, 136

Carleman linearization, 122, 124
cheap gradient principle, 92, 96, 278
coherent arithmetic, 34, 54, 92, 101, 108, 143, 144, 234, 263, 265
collateralized debt obligations (CDOs), 130
color code, 324, 329
computational fluid dynamics (CFD), 111, 114, 124
configuration interaction (CI) methods, 42, 43
conjugate gradient method, 118, 126, 139, 275, 293, 297
constraint satisfaction problems, 65, 253, *see also* satisfiability problem
contrastive divergence, 163, 167
convex optimization, 90–93, 134, 154, 279, 291, 299
coupled cluster (CC) methods, 13, 42, 43, 60, *see also* unitary coupled cluster (UCC) ansatz
credit valuation adjustments (CVAs), 130, 142
cryptocurrency, 107, 110
cryptography, 98, 169
  elliptic curve, *see* ECC
  post-quantum, 104, 106, 107, 131

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

public-key, 99
symmetric-key, 99, 108
curse of dimensionality, 123, 146
cutting-plane methods, 89

decoding (in QEC), 65, 316, 318, 321–324, 330
density functional theory (DFT), 42
dequantization, *see* quantum-inspired algorithms
derivative pricing, *see* option pricing
differential equations, 111–129
nonlinear, 114, 150
ordinary (ODEs), 113
partial (PDEs), 113
stochastic (SDEs), 122, 142
Diffie–Hellman, 99
Dijkstra's algorithm, 67
discrete logarithm problem, 102, 183, 226
elliptic curve, *see* ECDLP
distance (of a QEC code), 316
formula for, 330
of the surface code, 320
DMRG (density matrix renormalization group), 13, 42, *see also* tensor networks
DQC1-complete, 308
DQC1-hard, 177
duality (in optimization), 79, 84, 182, 183, 292, 302
dynamic programming, 67, 68, 259

Eastin–Knill theorem, 317, 329
ECC (elliptic curve cryptography), 99, 103, 106
ECDLP (elliptic curve discrete logarithm problem), 100, 105
effective field theory (EFT), 58
eigenstate filtering, 8, 18, 35, 233, 234, 272
electronic structure problem, 29–44, 61, 204, 248
element distinctness problem, 68
elliptic curve cryptography, *see* ECC
energy-based models, 160–169, *see also* Boltzmann machine
exact diagonalization, 13, 27, 42, 247

factoring, 100, 103, 227, 233, 285
Fermi–Hubbard model, 5–15, 328
finite difference methods (FDMs), 33, 112, 142, 277, 283
finite element method (FEM), 112, 121, 123, 126
finite volume method (FVM), 112
Fokker–Planck equation, 123

Gaussian elimination, 89, 118, 126, 138, 155, 157, 274, 293
Gaussian process regression, 151–154, 159
Gibbs sampling, 205, 228, 233, 243–249
applications of, 8, 18, 25, 37, 47, 86, 166, 302
dynamic, 80, 86, 304
of classical systems, 80, 82, 164, *see also* Monte Carlo methods, Markov chain
gradient descent, 93, 94, 97, 163, 164, 181, 279, 283
Greeks (in finance), 130, 141, 145, 280
Grover search, 62–68, 71, 86, 175, 259, 294, 300, *see also* amplitude amplification

Hamiltonian simulation, 202, 205, 209–224, 228, 264
applications of, 8, 18, 26, 37, 47, 53, 59, 95, 117, 122, 171, 192
as access model, 7, 34, 36, 59, 247
of adiabatic evolution, 72, 251, 294
Hamming weight phasing, 11, 20
Hartree–Fock, 42
heat equation, 112, 123
Heisenberg model, 17, 19, 21
hidden subgroup problem, 102, 227
high-performance computing (HPC), 60, 75, 109, 111, *see also* parallelism in classical implementations

initial value problem, 114
integer factorization problem, *see* factoring
integer program (IP), 134
mixed- (MIP), 68, 134, 136
interior point methods (IPMs), 81, 88, 138, 291, 301, *see also* quantum interior point methods
software implementation, 88
Ising model, 16, 21, 70, 161, 164, 252
iterative refinement, 274, 288

Jordan–Wigner transformation, 7, 25, 34

Kaczmarz method, 89, 139, 274, 297
Kaiser window, 37, 232
kernel methods, 151, 174, 181, *see also* quantum kernel methods
Kikuchi method, 171, 173

LABS (low autocorrelation binary sequence) problem, 72
Lagrange multipliers, 133, 135, 155, 292
lattice Boltzmann equation, 115, 122, 124
lattice gauge theories (LGTs), 53–56

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

lattice Schwinger model, 54, 56
lattice surgery, 263, 326
  resource estimate, 329–330
linear combination of unitaries (LCU), 195–197, *see also* block-encoding of an LCU
  applications of, 117, 245, 273
  method of Hamiltonian simulation, 218–221, 264
linear program (LP), 78, 83, 292, 299, 303
low-density parity check codes, 317, 329

magic state distillation, *see* state distillation
matrix product state (MPS), 7, 14, 22, 23, 35, *see also* tensor network methods
  tomography, 287
MAX-CUT, 70, 298
Maxwell's equations, 112, 114, 123
Metropolis–Hastings algorithm, 163, 243, 245
Monte Carlo methods
  estimating expectation values, 141, 241
  Markov chain (MCMC), 161, 163, 167, 243
  multilevel, 146
  quantum, *see* quantum Monte Carlo
  quantum-accelerated, 122, 131, 143–146, 165, 265
  quasi–, 146
  style of Gibbs sampling, 8, 20, 25, 90, 166, 233, 245–247
multiplicative weights update (MWU) method, 248, 299–304
  applications of, 79, 86, 136

Navier–Stokes equation, 111, 114, 122
neural networks, 122, 125, 150, 161, 174, 179, 323, *see also* quantum neural networks
nonconvex optimization, 94–97, 134, 136, 298
NP, 62, 66, 342
NP-complete, 20, 343
NP-hard, 68, 134, 166, 177, 247, 283, 307, 343
nuclear magnetic resonance (NMR), 16, 17, 188, 198

option pricing, 130, 140–147, 241
order finding, 100
overlap estimation, 241
  applications of, 26, 38, 118, 153, 156

parallelism
  in classical implementations, 104, 126, *see also* high-performance computing
  in decoding, 322
  in QRAM, 66, 256
  of classical methods, 74, 109, 110, 126, 146, 167, 258
  of quantum methods, 75, 95, 102, 103, 137, 242, 262, 268, 338
  of state distillation, 257, 328
parameterized quantum circuit, *see* variational quantum algorithms
partition function, 162, 244, 248, 309
phase estimation, *see* quantum phase estimation
Poisson equation, 113, 115
portfolio optimization, 68, 87, 131–140, 269, 297, 303
power method, 171, 176, 178
preconditioning (linear system), 119, 124, 126, 275
principal component analysis (PCA), 149, 169, 234, *see also* tensor PCA
product formulas, 211–216, 230, 252
  applications of, 7, 18, 26, 38, 47
  resource estimate for, 11, 12, 21, 40, 41, 54, 60, 74
projected entangled pair state (PEPS), 22, *see also* tensor network methods

QAOA (quantum approximate optimization algorithm), 69, 71, 75, 139, 173, 253, 284
qDRIFT, 216–218
QMA, 342
QMA-complete, 20, 22, 343
QMA-hard, 12, 13, 39, 177, 247, 343
QRAM (quantum random access memory), 65, 85, 150, 255–259, 303
  applications of, 67, 68, 80, 81, 86, 92, 117, 135, 136, 149, 155, 165, 193, 261, 273, 295
QROM (quantum read-only memory), 258
QSVT (quantum singular value transformation), 205–209, 229, 236, 242
  applications of, 73, 175, 221, 237, 245, 265, 271, *see also* eigenstate filtering
  method of Hamiltonian simulation, 222–224
quadratic program, 133, 154
quantum adiabatic algorithm (QAA), 69, 72, 76, 233, 250–254, 284
  applications of, 6, 18, 26, 36, 47, 72, 164, 165, 265, 272, 294
quantum annealing, 76, 134, 139, 165–168, 249, 252
quantum arithmetic, *see* coherent arithmetic
quantum eigenvalue transformation, 204, *see also* QSVT

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

quantum field theories, 51–57, 210, 214, 227
quantum Fourier transform (QFT), 225–227
  applications of, 101, 231, 277
  approximate, 102, 226
  generalized, 102, 227
  operator, 245
quantum gradient estimation, 241, 276–280, 288
  applications of, 10, 38, 86, 91, 95, 145, 166, 294
quantum interior point methods (QIPMs), 290–298
  applications of, 84, 135, 155
quantum kernel methods, 181–183
quantum key distribution, 98, 131
quantum linear system solver (QLSS), 205, 209, 228, 234, 253, 270–275, 284
  applications of, 84, 117, 135, 149, 152, 155, 293
quantum minimum finding, 63, 80, 86, 237
quantum Monte Carlo (QMC), 13, 20, 42, 44, 52, 55, 58, 75, 284
quantum neural networks, 180–181, 285
quantum phase estimation (QPE), 203, 205, 218, 227–234, 253
  applications of, 9, 18, 26, 36, 47, 59, 73, 101, 165, 171, 175, 239, 244
  consistent, 232
  unbiased, 232, 288
quantum random access memory, *see* QRAM
quantum signal processing (QSP), 198–201, 229, 236, *see also* QSVT
  applications of, 144, 233
  method of Hamiltonian simulation, 8, 18, 26, 38, 222–224, 230
  resource estimate for, 12, 21, 145
quantum simulated annealing, 165
quantum singular value transformation, *see* QSVT
quantum tomography, 148, 226, 241, 260, 274, 279, 286–290, *see also* shadow tomography
  applications of, 84, 118, 135, 291
quantum walks, 67–68, 211, 237
quantum-inspired algorithms, 48, 150, 153, 157, 158, 275, 305
qubitization, 201–205
  for Hamiltonian simulation, 12, 26, 38, 54, 222
  for QPE, 8, 9, 37, 231
  resource estimate for, with QPE, 11, 40, 41
QUBO (quadratic unconstrained binary optimization), 16, 70, 134, 140

rate (of a QEC code), 316
reaction-diffusion equations, 122
recommendation systems, 149, 234, 258
rejection sampling, 36, 165, 175, 233, 245, 246
reversible arithmetic, *see* coherent arithmetic
RSA (Rivest–Shamir–Adleman), 99, 103, 106

SAT, *see* satisfiability problem
satisfiability problem, 64, 69, 70, 74, 342
Schöning's algorithm, 64, 67
Schrödinger equation, 30, 45, 58, 95, 114, 123, 210, 251, 335
second-order cone program (SOCP), 83, 135, 154, 294, 297
semidefinite program (SDP), 83, 248, 288, 294, 297, 299, 303
shadow tomography, 289, 290
Sherrington–Kirkpatrick model, 68, 70, 74, 162, *see also* spin glass
Shor's algorithm, ix, 99–103, 228, 233
short-path algorithm, 73, 233
sign problem, 52, 56, 75
Simon's algorithm, ix, 102, 344
simplex method, 291
simulated annealing, 163, 167, *see also* quantum simulated annealing
single-shot QEC, 322, 329
spectral methods, 112, 115, 116, 123, 127
spin glass, 71, 162, 169
spin models, 15–23, 70, 161, 224
stabilizer code, 320
state distillation, 103, 257, 263, 327–330
state preparation
  adiabatic, *see* quantum adiabatic algorithm
  from classical data, 7, 88, 152, 155, 156, 158, 197, 259–265
  garbage-free, 263
  of ground states, 6, 12, 17, 26, 30, 53, 58, 164, 178, 229, 250
  of thermal states, *see* Gibbs sampling
  of trial/ansatz states, 7, 26, 35, 272
stoquastic, 75
supercomputer, *see* high-performance computing
superconductor, 5, 6
support vector machine, 154–157, 181, 297, 300
surface code, 103, 147, 263, 320–331
  resource estimate for, 65, 103, 329–330

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

SYK model, 24–28

tensor network methods, 13, 14, 20, 42, 56, 147, 158, 168, 226, 285, 305–309, 321
tensor PCA, 76, 169–173
threshold (in QEC), 316–318, 322
tomography, *see* quantum tomography
topological data analysis (TDA), 173–179, 205, 259
traveling salesperson problem, 64, 67, 68, 259
Trotterization, *see* product formula
turbulence (fluids), 122, 124

unitary coupled cluster (UCC) ansatz, 284

value at risk (VaR), 131, 142
  conditional (CVaR), 131
variational quantum algorithms (VQAs), 63, 96, 180, 252, 279, 281–285
  fault-tolerant applications of, 7, 35, 47, 71, 144, 164
  NISQ applications of, 14, 23, 44, 49, 57, 61, 105, 110, 128, 139
variational quantum eigensolver (VQE), 14, 44, 178, 283
Vlasov equation, 112

wave equation, 111

zero-sum game, 78–82, 300, 303

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39

Downloaded from https://www.cambridge.org/core. IP address: 189.217.92.64, on 12 Feb 2026 at 21:16:16, subject to the Cambridge Core terms of use, available at https://www.cambridge.org/core/terms. https://www.cambridge.org/core/product/EF7A52B88199BC0DA5A2CC99794A8C39